# UNIVERSIDAD POLITÉCNICA SALESIANA

**SEDE QUITO – CAMPUS SUR**

**CARRERA DE INGENIERÍA DE SISTEMAS**
**MENCIÓN TELEMÁTICA**

**TESIS PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SISTEMAS**

**Diseño e implementación de un sistema académico vía web para la Escuela Fiscal Mixta Ciudad de Ibarra de la parroquia de Guayllabamba**

Andrea del Carmen Mora Naranjo
María Elisa Hernández Quimbiulco

**DIRECTOR:** ING. XAVIER CALDERÓN

**FECHA: Julio, 2012**

## DECLARACIÓN

Nosotros, Andrea del Carmen Mora Naranjo y María Elisa Hernández Quimbiulco, declaramos bajo juramento que el presente trabajo de titulación es de nuestra autoría y que no ha sido previamente presentado para ningún grado o calificación profesional y, que hemos consultado las referencias bibliográficas.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual de este trabajo, a la Universidad Politécnica Salesiana, según lo establecido por la ley de propiedad intelectual, por su reglamento y por la normatividad institucional vigente.

_____ _____

ANDREA MORA MARÍA ELISA HERNÁNDEZ

## CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Andrea del Carmen Mora Naranjo y María Elisa Hernández Quimbiulco bajo mi dirección.

_____

Ing. Xavier Calderón

Director de tesis

## AGRADECIMIENTOS

Primero agradezco a Dios por todas las bendiciones que me ha dado durante toda mi vida, las cuales me han permitido llegar a cumplir con una meta tan importante.

Agradezco a las personas más importantes en mi vida, mis padres Hugo y Anita. Ellos que han sido las personas que siempre estuvieron preocupados por que tenga todas las herramientas para poderme defenderme en la vida.

También quiero agradecer a mi mejor amigo, Daniel quien ahora también es mi esposo por su apoyo y compañía durante todo este tiempo.

Agradezco a mi gran amiga y compañera de aulas, María Elisa con quien ha sido un gran placer trabajar en este proyecto.

Y por último al nuestro director de tesis Ing. Xavier Calderón quien con su apoyo y sus acertados consejos nos ha ayudado a culminar este proyecto.

**ANDREA MORA NARANJO**

## AGRADECIMIENTOS

En mi primer lugar deseo agradecerle a Dios por haberme brindado la fuerza y sabiduría para cumplir este objetivo en mi vida.

A mis padres por toda su dedicación, apoyo, cariño y la paciencia que me han brindado día a día no solo para cumplir con esta meta sino en el diario vivir.

Por todo el apoyo y la fuerza que aunque ya no esté presente me la continúa brindando Tío Marquito.

A mí querida amiga Andrea por todo el tiempo invertido para sacar adelante nuestro proyecto.

Por último a mi director de tesis Ing. Xavier Calderón por su guía y colaboración.

**MARÍA ELISA HERNÁNDEZ Q.**

## DEDICATORIA

Dedico esta tesis a toda mi familia ya que sin su cariño, constancia y paciencia me hubiese sido muy difícil cumplir la meta, ya que cada uno ha contribuido de una u otra manera en mi formación, a mi abuelito Eduardo a quien recuerdo con un inmenso cariño a mis hermanas Jenny y Anita por estar junto a mí, a mi esposo Daniel por su amor y dedicación y ahora incluso a mis sobrinos quien son un impulso para querer mejorar cada día.

Pero sobre todo a mis padres a quienes amo y admiro por todo el esfuerzo que han tenido que pasar para sacar adelante a su familia haciéndolo con todo el cariño del mundo.

**ANDREA MORA NARANJO**

## DEDICATORIA

La culminación de este proyecto está dedicado a Dios por guiarme y estar presente en mi vida en todo momento.

A mis queridos padres Armando y Francisca por todo su amor, paciencia y apoyo constante a lo largo de mi vida personal como en mi formación académica por ser la fuerza y el ejemplo que me llevan cada día a dar un paso más en mi vida, a mi querida hermana Carolina por su comprensión y cariño.

A mi Tío Marquito por todo su cariño, protección y apoyo incondicional para mi vida.

**MARÍA ELISA HERNÁNDEZ Q.**

# INTRODUCCIÓN

Con los avances tecnológicos e informáticos toda institución sin importar su índole se ve en la necesidad de simplificar procesos y brindar las mejores herramientas a su personal para un trabajo más eficiente.

Las instituciones educativas no se escapan de esta realidad ya que también manejan procesos complejos que necesitan ser controlados y reducir al máximo el nivel de errores cometidos durante la ejecución de dichos procesos, es por esto que una aplicación web sería la solución para varios de los inconvenientes que se pueden presentar mediante la ejecución de cualquier proceso puesto que facilita el trabajo a distancia, no se requieren complicadas combinaciones de hardware/software, solo un computador y un navegador web.

Por estos motivos se crea un Sistema web para la institución “Escuela Fiscal Mixta Ciudad de Ibarra” con el fin de simplificar ciertos procesos que en la actualidad se realizan manualmente, sin ningún tipo de seguridades para la información, por lo que el sistema ofrece la flexibilidad de determinar niveles de acceso ó perfiles según el tipo de usuario, manteniendo la confidencialidad de los datos.

## RESUMEN

Este proyecto de tesis se crea como solución a varios de los problemas que se han presentado en la “Escuela Fiscal Mixta Ciudad de Ibarra”. Dicha solución es un sistema web desarrollado en Java con una base de datos MSQL y utilizando las tecnologías JPA (*Java Persistence API*), AJAX (*Javascript asíncrono Y Xml*) y JSF (J*avaserver Faces*).

Este proyecto se diseño basado en la metodología en cascada según la cual podemos dividir el proyecto en cinco partes.

**PARTE I**

Se describe el planteamiento del problema, objetivos, alcance y justificación del proyecto.

**PARTE II**

Análisis: en esta fase se identifica los requerimientos de usuarios y cada una de las tecnologías y metodologías utilizadas en todo el desarrollo del proyecto.

**PARTE III**

Diseño: En esta etapa se realizó el diseño de procesos, base de datos e interfaces.

**PARTE IV**

Implementación: en este capítulo se muestra los requerimientos básicos para el funcionamiento del sistema. Y las pruebas realizadas al sistema educativo.

**PARTE V**

Mantenimiento: En esta fase se describe los tipos de mantenimiento que se ofrecen a la institución que hará uso del sistema.

## CONTENIDO

## CONTENIDO DE TABLAS

## CONTENIDO DE FIGURAS

# CAPÍTULO I

## 1. ANTECEDENTES

Las instituciones educativas manejan gran cantidad de datos los cuales deben ser debidamente manipulados para evitar futuros errores, conservando la integridad de los mismos, por ello es necesario proporcionar tanto al personal docente como administrativo herramientas que agiliten y faciliten los diferentes procesos académicos.

Actualmente en la Escuela Fiscal Mixta Ciudad de Ibarra de la parroquia de Guayllabamba, no cuentan con un sistema académico que les permita mantener su información de manera ordenada, rápida y segura.

Razón por la cual todos los procesos como la matriculación, registro de notas, notificaciones al personal se realizan de manera manual.

Para el proceso de matriculación se asigna un período establecido por el Ministerio de Educación, en el cual un docente es escogido de manera aleatoria y asignado para realizar la matriculación de cada año de educación básica.

Las calificaciones de cada alumno se guardan en registros manuales almacenados en el archivo de la Institución. Para la entrega de las calificaciones a los alumnos se compran las respectivas libretas en el Ministerio de Educación y se procede a llenar las mismas.

Las notificaciones al personal docente se las realiza mediante convocatorias, reuniones o notificaciones en la cartelera de la Institución con los respectivos datos informativos de las mismas.

## 1.1 OBJETIVOS

### 1.1.1 OBJETIVO GENERAL

Desarrollar e implementar una aplicación web que permita gestionar la información de los procesos educativos y administrativos de la Escuela Fiscal Mixta Ciudad de Ibarra de la parroquia de Guayllabamba a través de modernas tecnologías que ofrezcan resultados seguros, rápidos y eficientes.

### 1.1.2 OBJETIVOS ESPECÍFICOS

- Dar a conocer las características, atributos y propiedades de las herramientas a ser utilizadas para el desarrollo de la aplicación.

- Implementar una herramienta que cumpla con los requerimientos que los usuarios de la institución soliciten y así, permita desempeñar las funciones cotidianas con agilidad y facilidad.

- Generar información procesada, mediante reportes, para un seguimiento por parte del usuario.

- Resguardar la integridad de la información a través del manejo de perfiles.

## 1.2 JUSTIFICACIÓN

El diseño e implementación de un sistema académico vía web para la Escuela Fiscal Mixta Ciudad de Ibarra de la parroquia de Guayllabamba se justifica en los siguientes puntos:

1. La importancia de implementar un sistema de gestión académica nace con la necesidad de manejar adecuadamente la información en toda institución educativa para obtener resultados rápidos, fáciles y seguros.

2. Instituciones educativas, autoridades, docentes deben manejar herramientas que les permita realizar sus actividades con mayor facilidad de acuerdo al rol que desempeñan dentro de la institución, esto se puede lograr mediante un sistema de gestión académica que utilice tecnologías innovadoras y confiables.

3. La ausencia de tecnología en algunas instituciones obliga a manejar la información a través de papeles ó archivo poniendo en riesgo la misma, lo que genera un mayor gasto a la institución y contamina al medio ambiente.

4. Tecnificar los procesos existentes en la institución educativa reduciendo así el margen de error en los mismos.

5. Obtener a través del software académico documentos e información verídica.

## 1.3 DELIMITACIÓN

Diseño e implementación de un software eficaz que satisfaga necesidades tanto del personal administrativo como docente, mediante módulos que den solución a los requerimientos de dicho personal:

**Módulo de administrador**

Este módulo contendrá la parte de administración del sistema, usuarios, roles, administración de periodos, auditorias, acceso a datos.

**Módulo de parámetros**

Este módulo está dedicado al personal administrativo en donde se podrá fijar parámetros utilizados para los diferentes procesos como son asignaturas, cursos, paralelos etc.

**Módulo de matriculación**

Este será otro módulo dedicado al personal administrativo ya que en él se realizaran inscripciones, matriculas.

**Módulo de calificaciones**

En este módulo se permitirá a los docentes ingresar las calificaciones de los estudiantes, siendo este muy didáctico para el mismo.

**Módulo de reportes**

En este módulo el usuario podrá obtener varios reportes como los siguientes:

- Reporte de Personal docente.
- Reporte de personal Administrativo.
- Reporte de estudiantes matriculados.
- Reportes de calificaciones por nivel.

- Reportes de promedios por cursos.

- Reporte de calificaciones por estudiante.

- Reporte de lista de estudiantes por nivel.

## 1.4 ESTRUCTURA FUNCIONAL DE INSTITUCIÓN

Existen 9 profesores incluyendo el Director de la Institución que realiza labores administrativas y cumple con el rol de profesor, de la misma manera los docentes de la Institución realizan las labores administrativas en diferentes turnos para cubrir las necesidades de la “Escuela Fiscal Mixta Ciudad de Ibarra” sin descuidar su rol de profesor.

Actualmente la institución consta de 260 estudiantes, 7 niveles de educación básica los mismos que tienen un paralelo por cada año escolarizado a excepción de cuarto año que posee dos paralelos y se encuentran distribuidos de la siguiente manera:

- Primero año de educación básica: 29 alumnos.

- Segundo año de educación básica: 28 alumnos.

- Tercer año de educación básica: 33 alumnos.

- Cuarto año de educación básica: 25 alumnos.
- Cuarto año de educación básica: 27 alumnos.
- Quinto año de educación básica: 40 alumnos.
- Sexto año de educación básica: 40 alumnos.
- Séptimo año de educación básica: 38 alumnos.

En base a la información obtenida de la “Escuela Fiscal Mixta Ciudad de Ibarra” se establece la siguiente estructura funcional para los docentes:

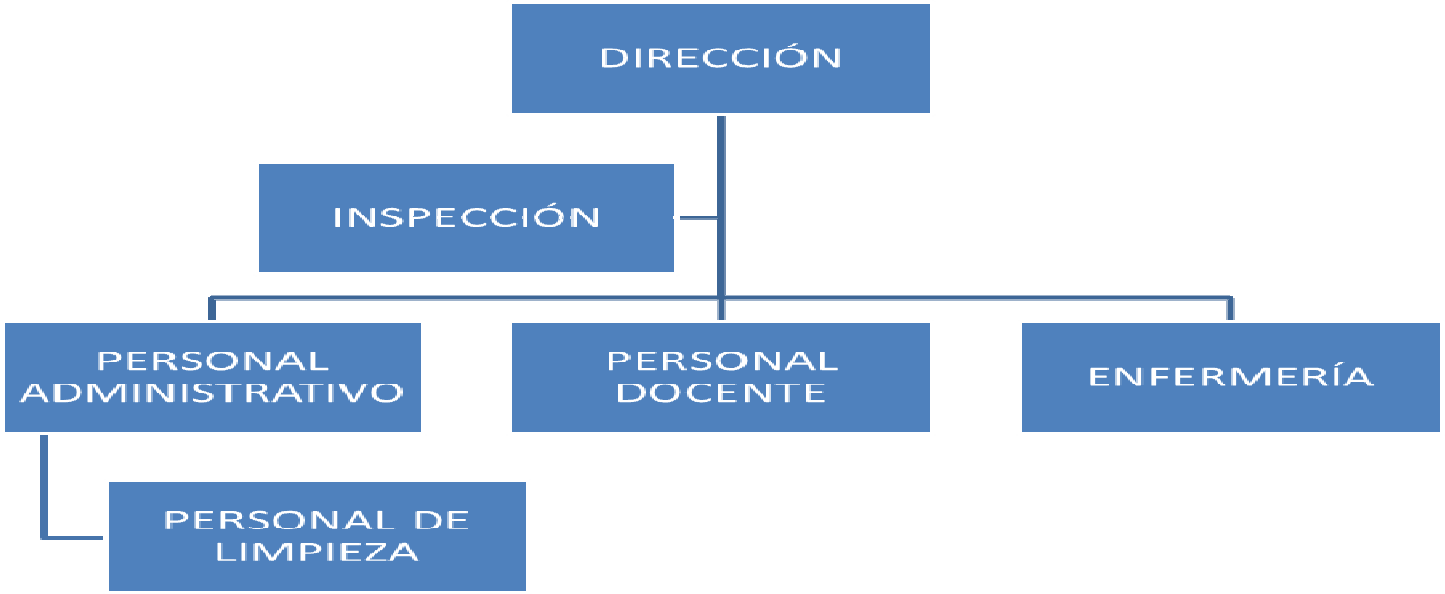



Fig 1.1 Estructura Funcional

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

La institución tiene un laboratorio el cual es utilizado tanto por los estudiantes como por el personal docente**.**

El laboratorio posee 12 computadoras interconectadas con topología estrella mediante un *swich*, A cada equipo tiene asignado una dirección *IP* por medio del "*Protocolo de Internet TCP/IP*".

La red pose un direccionamiento IP estático  ya que es una red LAN pequeña
Equipo Nº 1 Dirección IP: 192.168.0.1 Máscara de subred: 255.255.255.0

Equipo Nº 2 Dirección IP: 192.168.0.2 Máscara de subred: 255.255.255.0
.
.
.
.
Equipo Nº 12 Dirección IP: 192.168.0.12 Máscara de subred: 255.255.255.0

En la dirección también se encuentra una computadora la misma que no está en red y funciona para redactar documentos, oficios para ser entregados a los docentes, estudiantes, Ministerio de educación según sea la necesidad.

# CAPÍTULO II

## 2. MARCO TEÓRICO

### 2.1 CONCEPTOS UTILIZADOS PARA EL DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA

#### 2.1.1 INTRODUCCIÓN

Para el desarrollo del sistema vía web, se ha elegido tecnologías que gracias a sus características harán que dicho sistema sea estable, confiable y amigable con el usuario.

Para el manejo de datos se ha elegido utilizar persistencias ya que permiten manipular los datos de la base utilizando objetos, liberando al desarrollador del manejo manual de los datos, con apenas un ligero incremento en el tiempo de ejecución.

La utilización de AJAX (*Javascript asíncrono y Xml*) permite realizar cambios sobre las páginas sin la necesidad de que cada vez que se haga una petición deba cargarse toda la página, significando aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

A continuación se describe más a fondo la metodología, arquitectura y herramientas utilizadas en el sistema.

### 2.1.2 APLICACIONES WEB

Una aplicación web es aquella a la que se puede acceder mediante un cliente web es decir un computador que posea un navegador de internet, se codifica en un lenguaje soportado por los navegadores web permitiendo acceder a través de internet en cualquier parte del mundo.

### 2.1.3 APACHE TOMCAT 6.0

Es un servidor web que soporta *servlets*[1] y JSP *(JavaServer Pages)*, po*see* un compilador que transforma páginas con extensión *".jsp"* en *servlets.* Está descrito en lenguaje *Java* lo que permite funcione sobre cualquier plataforma que posea una máquina virtual Java.

Las características que ofrece está herramienta son:

- Constante desarrollo, para mejorar su funcionalidad.

---

[1] La palabra ***SERVLET*** deriva de otra anterior, *applet*, que se refería a pequeños programas que se ejecutan en el contexto de un navegador web. Por contraposición, un *servlet* es un programa que se ejecuta en un servidor.

- Funciona sobre cualquier plataforma.

- Apache Tomcat 6.0 funciona como servidor web y de aplicaciones.

- Posee un entorno con alto nivel de disponibilidad y tráfico.

### 2.1.4 NETBEANS 6.7

*NetBeans 6.7* es un entorno de desarrollo integrado, modular y normalizado escrito en lenguaje java. La plataforma de esta herramienta permite desarrollar por módulos a partir de un conjunto de componentes prediseñados para el programador permitiendo realizar modificaciones a los mismos según sea la necesidad.

Las principales características de *NetBeans 6.7* son:

- Permite obtener plug-ins para añadir funcionalidades a la herramienta.

- Es de código abierto.

- Software libre por lo cual es gratuito.

- Se encuentra en constante crecimiento, entregando mejoras a sus usuarios.

- Su plataforma permite la administración de usuarios, administración de configuración de usuarios, etc.

### 2.1.5 JSP (JAVASERVER PAGES)

Es una tecnología que permite de forma simplificada y rápida la creación de sitios web dinámicos.

El JSP es, ni más ni menos que código HTML, XML combinado con código Java.

Los JSP nos facilitan el desarrollo multicapa ya que la arquitectura del mismo es de tres capas.

La página JSP es una página web realizada con HTML común y código Java. La extensión de fichero de una página JSP es ".jsp" lo que le indica al servidor que esta página requiere un manejo especial.

### 2.1.5.1 ENTORNO DE EJECUCIÓN EN JSP

Para ejecutar las páginas JSP se llama a una página con extensión ".jsp", la misma que será ejecutada por un motor JSP en un *Servlet* de java. El *servlet* es manejado por un motor de *servlet* el mismo que usa un cargador de clases y lo ejecuta para crear un html dinámico y enviarlo al navegador como se muestra en la Fig 2.2 Ejecución de JSP.

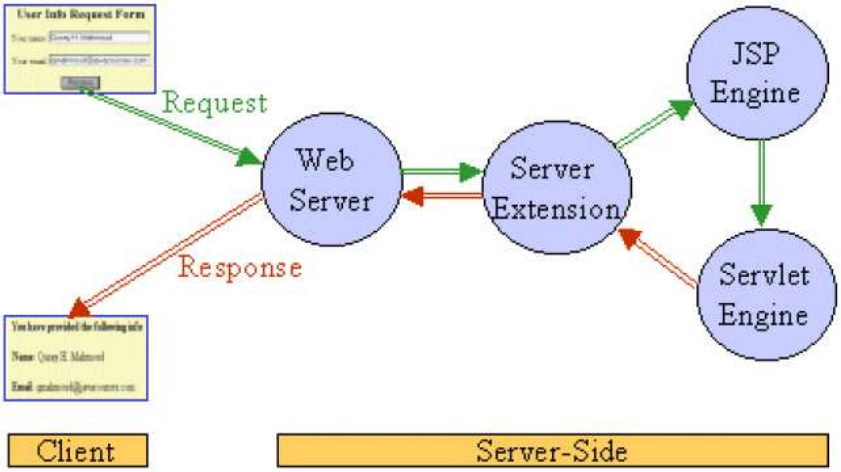



**Fig 2.1** Ejecución de JSP

FUENTE wiki.icesoft.org/display/ICE/JSP

#### 2.1.5.2 VENTAJAS DE JSP

- Permite crear aplicaciones web que se ejecuten en varios servidores con diferente plataforma, puesto que JSP es en esencia un lenguaje multiplataforma.

- Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar *scripts* ejecutables en el servidor, en código Java. Por lo tanto, las páginas JSP se pueden codificar con cualquier editor HTML/XML.

### 2.1.6 JSF (JAVASERVER FACES)

JSF es un *framework* creado para construir interfaces de usuario para aplicaciones web, ofrece un marco de trabajo que facilita el desarrollo de aplicaciones, separando las diferentes capas de una arquitectura: presentación, negocio y datos.

#### 2.1.6.1 CARACTERÍSTICAS JSF

- Añade validación del lado del servidor y la conversión de datos.

- Define la navegación entre páginas.

- Conecta eventos generados en el cliente a código de la aplicación en el lado del servidor.

- Mapea componentes UI (Interfaz de usuario) a una página de datos del lado del servidor.

#### 2.1.6.2 PARTES DE UNA APLICACIÓN JSF

Una aplicación JSF contiene las siguientes partes:

- Posee un conjunto de páginas JSP.

- Necesita un conjunto de componentes *backing beans*, los cuales definen las propiedades y funciones para los componentes UI (Interfaz de usuario) sobre una página JSP.

- Archivos de configuración de la aplicación, los cuales definen las reglas de navegación y configuración de *beans* u otros objetos como componentes definidos por el cliente.

- Un archivo descriptor de despliegue (web.xml).

- Conjunto de objetos cliente creados por el desarrollador de la aplicación como: componentes, validadores, conversores o *listeners*.

- Un conjunto de tags del cliente para representar los objetos del cliente sobre la página.

#### 2.1.6.3 VENTAJAS DE UTILIZAR JSF

JSF posee las siguientes ventajas:

- Una clara separación entre vista y modelo.
- Desarrollo basado en componente, no en peticiones.
- Las acciones del usuario se ligan muy fácilmente al código en el servidor.
- Creación de familias de componentes visuales para acelerar el desarrollo.
- Manejo de elementos UI (Interfaz de usuario) como objetos con estado en el servidor.
- Estandarización: Los más grandes vendedores de herramientas de desarrollo colaboran con su mantenimiento.

Es necesario tomar en cuenta que JSP y JSF son diferentes ya que muchas de las veces se piensa que son una versión mejorada el uno del otro por lo que hay que tomar en cuenta estos dos puntos:

- JSP (*Java Server Pages*) son las páginas web dinámicas de java; dinámicas ya que se puede obtener contenido de una base de datos.

- JSF (*Java Server Faces*) son componentes que se utilizan en las paginas JSP para facilitar el desarrollo al momento de diseñar formularios o mostrar datos que se encuentren en una base.

### 2.1.7 AJAX (*JAVASCRIPT* ASÍNCRONO Y *XML*)

Es una técnica para el desarrollo de aplicaciones web que se ejecuta en el lado del cliente es decir el navegador web, pero por debajo mantiene una comunicación asíncrona con el servidor.

Como es asíncrona si se desea realizar algún cambio en la aplicación, no será necesario que la página se cargue de nuevo, lo que permite tener aplicaciones de tipo interactivas, rápidas y eficientes, ya que los datos adicionales se requieren en el servidor y estos se cargan en un segundo plano sin interferir en el comportamiento de la página.

#### 2.1.7.1 VENTAJAS DE UTILIZAR AJAX

- Utiliza tecnologías ya existentes, haciendo posible la construcción de sitios web de manera rápida y dinámica.

- Soporta la mayoría de navegadores web.
- El usuario no tiene que esperar hasta que lleguen los datos del servidor.

- Portabilidad (no requiere plug-in como Flash).

- Mayor velocidad; esto debido a que no hay que retornar toda la página nuevamente.
- La página se asemeja a una aplicación de escritorio.

#### 2.1.7.2 MOTOR DE AJAX

El motor AJAX (*AJAX engine*) colocado entre el usuario y el servidor web evita el ciclo de recarga característico de las aplicaciones web tradicionales siendo así un fichero *JavaScript* que acompaña al HTML. El cual es cargado al inicio de la sesión y tiene la responsabilidad:

- Generar la interfaz visualizada por el usuario.

- Comunicarse con el servidor en representación del usuario, lo cual ocurre de manera asíncrona evitando que el espere demasiado cada vez que realice una acción.

#### 2.1.7.3 ¿CÓMO TRABAJA AJAX?

Toda acción realizada por parte del usuario que normalmente generaría una petición HTTP, adquiere la forma de una llamada *JavaScript* al motor AJAX. Cualquier acción que realice el usuario que no requiera un viaje al servidor como por ejemplo: validación de datos, edición de datos en memoria e incluso

la propia navegación, es procesada por el mismo. Si el motor necesita algún tipo de proceso en el servidor para generar la respuesta como enviar datos para procesamiento, carga de código de interfaz adicional o petición de datos nuevos, realiza las peticiones de manera asíncrona, normalmente usando el objeto *XMLHttpRequest*, sin detener la interacción del usuario con la aplicación.

#### 2.1.7.4 JSF CON AJAX

El *framework* para aplicaciones web “JSF” permite construir una biblioteca de componentes reutilizables, personalizados, ayuda a encapsular el interior de un componente, permitiendo ocultar el código complejo a los usuarios finales sin embargo esta gran herramienta lanza muchas peticiones al servidor, pero al complementarse con AJAX una tecnología que permite hacer una petición a una URL sin refrescar la página actual del navegador web, provee al usuario una sensación de aplicación de escritorio, por esto se optimiza dicho problema lo que lleva a realizar implementaciones de JSF con AJAX y permite:

- Actualizar los componentes en el navegador del usuario de manera selectiva, sin necesidad de recargar la página completa.

- La combinación JSF AJAX ofrece a las páginas un gran dinamismo sin complicar el desarrollo, evitando el uso de *javascript* codificado a mano, asegurando un mayor soporte a los navegadores web.

### 2.1.8 MYSQL

*MySql* es un motor de base de datos relacional, con un diseño multihilo que soporta una gran carga de datos de manera eficiente, posee un amplio sublenguaje *SQL (Structured Query Language).* Permite el uso de diferentes lenguajes de programación ya que es existe infinidad de librerías para su uso.

Es poseedor de varias características como las siguientes:

- Gestor de base de datos de software libre.
- Rapidez y facilidad en su uso.
- Es compatible con los principales lenguajes de programación como *Java.*
- Gestión de usuarios y contraseñas, manteniendo un alto nivel de seguridad.

### 2.1.9 JPA (JAVA PERSISTENCE API)

El objetivo del diseño de *Java Persistence API*, es no perder las ventajas de la orientación a objetos al interactuar con una base de datos.

Es fundamental conocer el funcionamiento de las tecnologías que se utilizan en JPA ya que afectará directamente al rendimiento de la aplicación. La idea es aplicar todos los recursos que maneja o implementa aplicaciones ORM (Mapeo objeto-relacional)[2] y de esta manera lograr mantener la independencia de la base de datos, bajo la lógica entre negocio y persistencia; orientarse en los procesos y lógica de la aplicación, mas no de conexiones e ingreso de datos a la base.

#### 2.1.9.1 VENTAJAS E INCONVENIENTES

Algunas de las ventajas de JPA consisten en que objetos transitorios pueden ser almacenados en forma permanente en una base de datos relacional.

Además, el uso de de JPA pretende reducir tiempos en la etapa de desarrollo, generando eficiencia y solidez en dicho proceso; permitiendo a los programadores enfocarse a resolver la lógica y problemática que maneja la línea de negocio, en lugar de ocupar su tiempo con el código de infraestructura.

---

[2] ORM: es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional

El *Java Persistence API* se basa en las mejores ideas de tecnologías como la persistencia *Hibernate*[3], *TopLink*[4], y JDO (*Java Data Objects*). Los clientes ya no tienen que enfrentarse a las incompatibilidades de modelos no estándar. Además, la *Java Persistence API* se puede usar tanto en entornos Java SE (*Java Platform, Standard Edition),* como en Java EE (*Java Platform, Enterprise Edition*).

No obstante, tiene la desventaja de que la implementación del proveedor se encuentra oculta a la aplicación lo cual la hace menos flexible.

Muchas de las características de la persistencia de otros *framework* se incorporaron a *Java Persistence API*, y proyectos como *Hibernate* y *TopLink* son ahora implementaciones de JPA.

#### 2.1.9.2 MODO DE EMPLEO

El proceso que ejecuta *framework* ORM para gestionar la interacción de las entidades con la JPA es:

Una aplicación que utiliza la interfaz *EntityManager*, esta interfaz proporciona métodos que realizan las funciones comunes de una base de datos con toda la

---

[3] *Hibernate* es una herramienta de Mapeo objeto- para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos

[4] *TopLink* es un *framework* utilizado para soportar la persistencia de datos.

estructura y definiciones CRUD (Crear, Obtener, Actualizar y Borrar) se encuentran desarrolladas en la clase administrador dentro de la aplicación.

Todas las entidades que pueden ser procesadas por un *EntityManager* se definen en una unidad de persistencia; que a su vez controla todas las operaciones de persistencia en la aplicación.

La unidad de persistencia está configurada por un contenedor extensión .XML de archivo llamado persistence.xml. Dicho archivo también define el *datasource*, el tipo de transacciones utilizadas por la aplicación, junto con otra información adicional como el puerto de conexión a la base de datos, usuario y contraseña. Estos ficheros se deben empaquetar en un .jar en el caso de ser una aplicación de escritorio o .war en el caso de ser aplicación web.

#### 2.1.9.3 ARQUITECTURA DE JPA

La Fig. 2.2 Arquitectura de una persistencia, muestra la relación entre los diferentes componentes de la arquitectura de JPA.

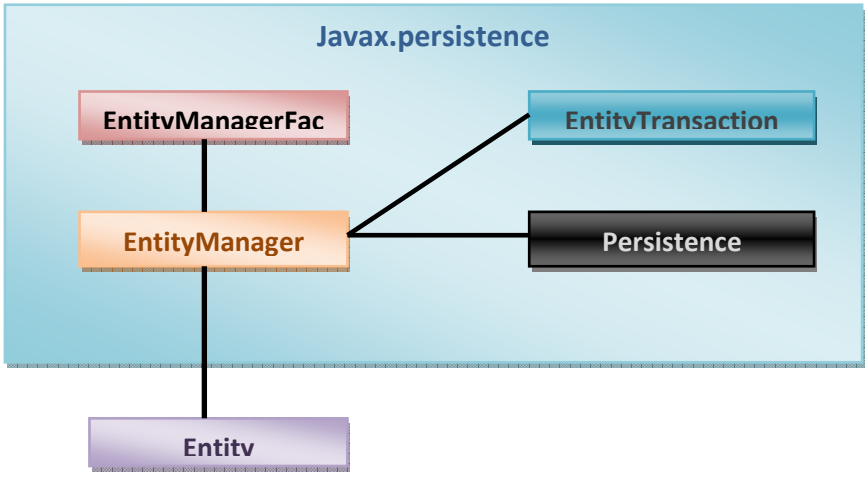



**Fig 2.2** Arquitectura de una Persistencia

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández.

***Persistence:*** La clase *javax.persistence.Persistence* contiene los métodos estáticos de ayuda para obtener una instancia de *EntityManagerFactory* de una forma independiente.

***EntityManager*:** La clase javax.*persistence.EntityManager* es la interfaz principal de JPA utilizada para la persistencia de las aplicaciones. Cada *EntityManage*r puede realizar operaciones CRUD (Crear, Obtener, Actualizar y Borrar).

***EntityManagerFactory:*** La clase javax.*persistence.EntityManagerFactory* nos permite crear objetos de EntityManager.

***Entity:*** La clase *javax.persistence.Entity* es una anotación Java que se coloca a nivel de clases Java serializables y que cada objeto de una de estas clases anotadas representa un registro de una base de datos.

***EntityTransaction****:* Relación 1 a 1 entre *javax.persistence.EntityTransaction* y cada instancia de *EntityManager* de *javax.persistence.EntityTransaction*, permite operaciones sobre datos persistentes de manera formen una unidad de trabajo transaccional, de esta manera llevamos a cabo la lógica de transaccionabilidad de base de datos para mantener la integridad de la información; así ejecuta bloques de sentencias y en el caso de llevar a cabo alguna transacción deja en su estado original a la base de datos haciendo un reproceso conocido como *RollBack*.

***Query:*** La interface *javax.persistence.Query* está implementada por cada desarrollador de JPA para encontrar objetos utilizando criterios propios de búsqueda. JPA estandariza el soporte para consultas utilizando Java *Persistence Query Language* (JPQL) y *Structured Query Language* (SQL).

Tomando en cuenta los conceptos anteriores; una entidad es un objeto de dominio de persistencia. Normalmente una entidad representa una tabla en el modelo de datos relacional y cada instancia de esta entidad corresponde a un registro en esa tabla. El estado de persistencia de una unidad se representa a través de campos persistentes o propiedades persistentes. Estos campos o propiedades usan anotaciones para el mapeo de estos objetos en el modelo de base de datos.

#### 2.1.9.4 TIPOS JAVA PERSISTENCE API

En el ámbito de desarrollo y lenguajes de programación existen 3 tipos de persistencias que son:

- Persistencia en Memoria.
- Persistencia de Aplicación.
- Persistencia de Objetos.

Para el sistema académico, la persistencia que se maneja en función de una estructura o *framework* ORM será Persistencia de Objetos.

#### 2.1.9.5 PERSISTENCIA EN MEMORIA

Se encuentra definida por la capacidad que tiene un objeto para seguir existiendo tras determinadas operaciones; es decir mantenerse vigente o cargado pese a otros procesos simultáneos mientras se mantenga la referencia de dicho objeto. La operación más común que se presta a la persistencia en memoria es la asignación.

#### 2.1.9.6 PERSISTENCIA DE APLICACIÓN

Esta dada por la capacidad de mantener vigente la información en tiempo de ejecución del programa o aplicación; es decir se mantiene en memoria RAM (*Random access memory*) el tiempo que ha tomado dichos procesos.

#### 2.1.9.7 PERSISTENCIA DE OBJETOS

La persistencia de objetos está definida por la inicialización de objetos con sus atributos por defecto y el posterior almacenamiento a través de la utilización de *frameworks* ORM.

### 2.1.10 METODOLOGÍA DE INVESTIGACIÓN PARA EL DESARROLLO DEL SISTEMA

#### 2.1.10.1 METODOLOGÍA EN CASCADA

La metodología en cascada o modelo en cascada establece un orden riguroso para las etapas del proceso de desarrollo de software de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior.

Esta metodología señala que ningún proceso puedo comenzar sin antes haberse terminado el anterior, de esta forma se asegura que cualquier error detectado en una etapa anterior pueda ser corregida y rediseñada.

De manera esquemática las fases del modelo cascada son:

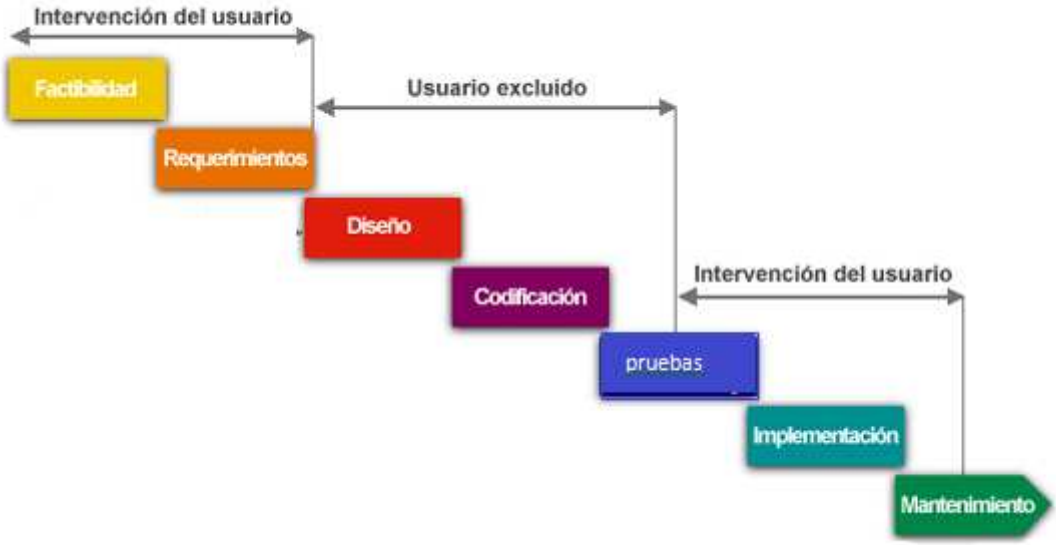



**Fig 2.3** Modelo Cascada

FUENTE: Tesista(s)
FUENTE: wiki/Desarrollo_en_cascada

En cada una de las fases se puede introducir más detalle si en una etapa superior se aprecia la necesidad de aquello, de esta manera se obtiene un código ejecutable y eficiente que incorpore los requisitos necesarios.

# CAPÍTULO III

## 3. ANÁLISIS

### 3.1 ETAPA ANÁLISIS DE REQUERIMIENTOS

En esta fase se obtienen y analizan las necesidades de los usuarios que utilizarán el sistema, para determinar los objetivos que se deberá cumplir.

Esta es la primera fase del proyecto y la más importante ya que aquí en donde se trabaja conjuntamente con el usuario para obtener toda la información necesaria para el desarrollo del sistema ya que será con él con quien se establecerán los procesos del sistema.

En esta etapa fue muy importante interactuar de manera correcta con el usuario ya que muchas de las ocasiones no se sabe que es lo que quiere lograr con un sistema informático o lo que puede pedir para mejorar su trabajo así que se debe dar ideas y llegar a la conclusión de lo que se va a realizar.

La manera en que se obtuvieron los requerimientos para desarrollar el sistema fue a través de una entrevista con los docentes de la Institución puesto que

ellos manejan los procesos de la misma, con lo cual se obtuvo la mayor parte de información, y se aplicó una encuesta para despejar ciertas dudas.

**Véase Anexo1**.

En base a las encuestas realizadas a los docentes de la Institución, se obtuvieron los siguientes resultados; de los cuales se desprenderán los requerimientos para realizar el sistema académico de la Escuela Fiscal Mixta Ciudad de Ibarra.

| **1.- ¿Cuántas personas realizan las tareas administrativas?** | **Administrativos** |
|---|---|
| De 1 a 10 | 1 |
| Más de 10 | 0 |
| Menos de 10 | 0 |

Tabla 3.1. Personas a realizar tareas administrativas

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

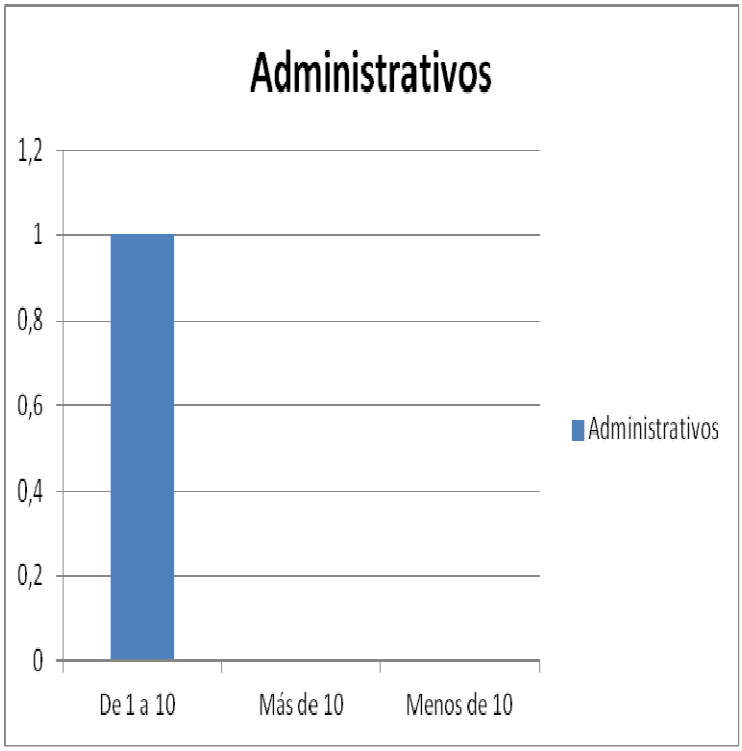



**Fig 3.1** Personas a realizar tareas administrativas

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

| 2.- ¿Cuál es el número de docentes? | # Docentes |
|---|---|
| De 1 a 10 | 9 |
| Más de 10 | 0 |
| Menos de 10 | 0 |

Tabla 3.2. Número de docentes

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

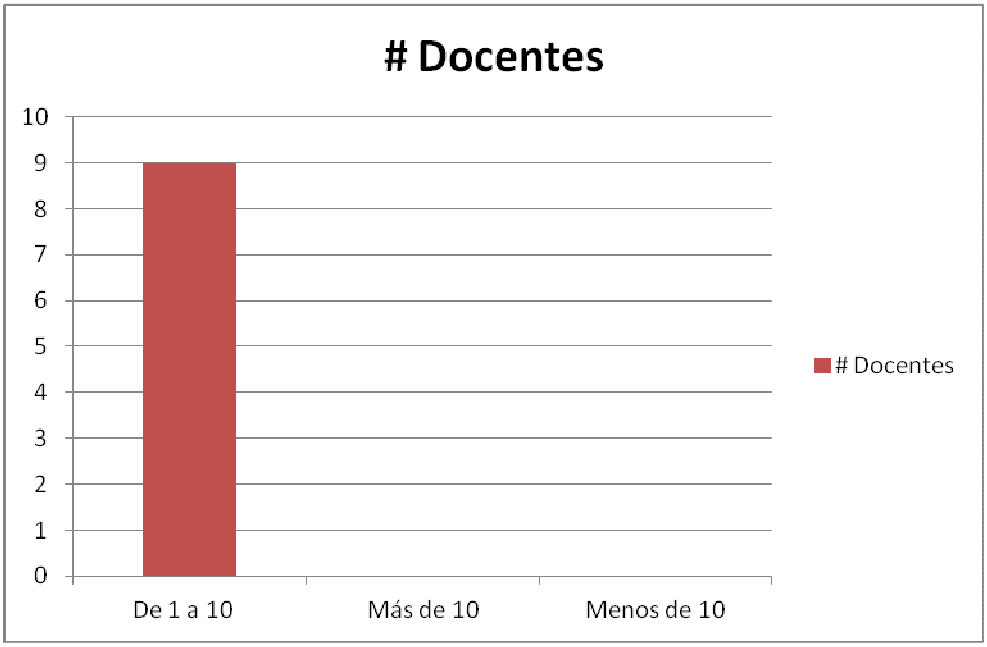



**Fig 3.2** Número de docentes

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

| 3.- ¿Cómo es el proceso para entregar el reporte de calificaciones trimestralmente? | Calificaciones |
|---|---|
| **Libretas (Manualmente)** | 9 |
| **Hojas impresas (Automático)** | 0 |

Tabla 3.3. Proceso entrega de calificaciones

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

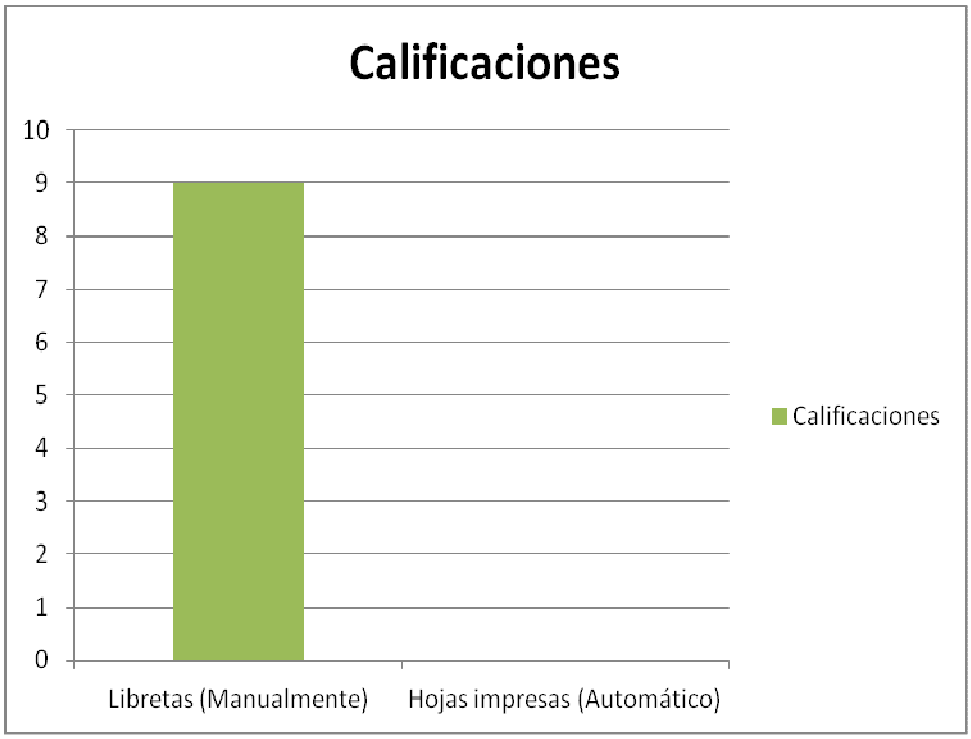



**Fig 3.3** Proceso entrega de calificaciones

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

| 4.- ¿Cómo es el proceso para entregar cualquier otro tipo de reporte administrativo? | Reporte Administrativo |
|---|---|
| A través de un sistema académico (Automático) | 9 |
| Por medio de formatos establecidos (Máquina de escribir) | 0 |

Tabla 3.4. Entrega de reportes administrativos

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

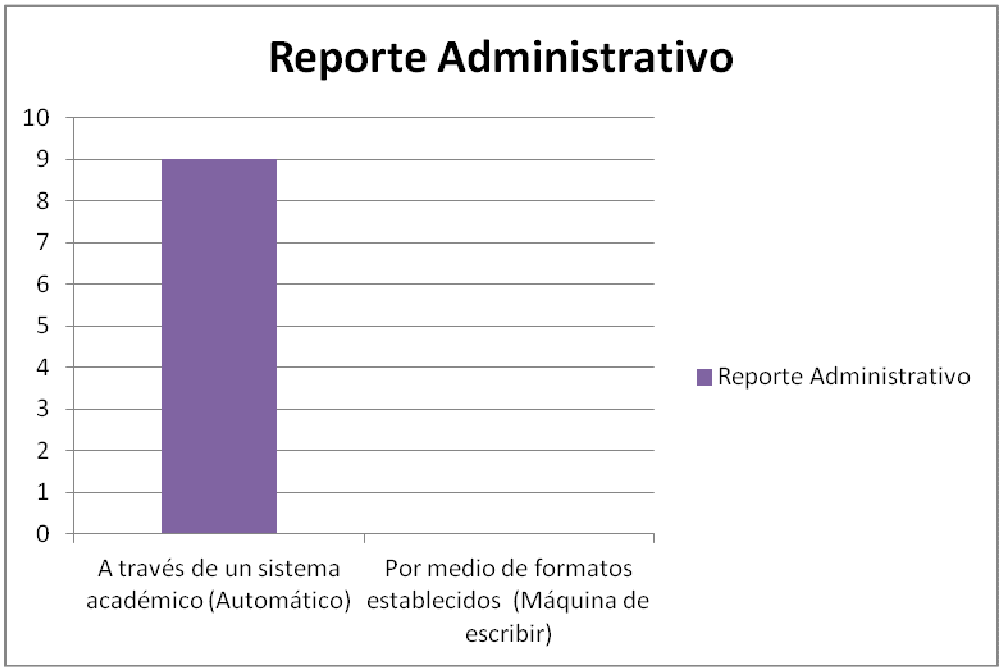



**Fig 3.4** Entrega de reportes administrativos

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

| 5.- ¿Qué proceso realiza el docente para pasar las notas al personal administrativo? | Entrega de Notas |
|---|---|
| **Hojas de cálculo** | 0 |
| **Hojas llenadas a mano** | 9 |
| **Vía web** | 0 |

Tabla 3.5. Proceso de entregar de notas al personal administrativo

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

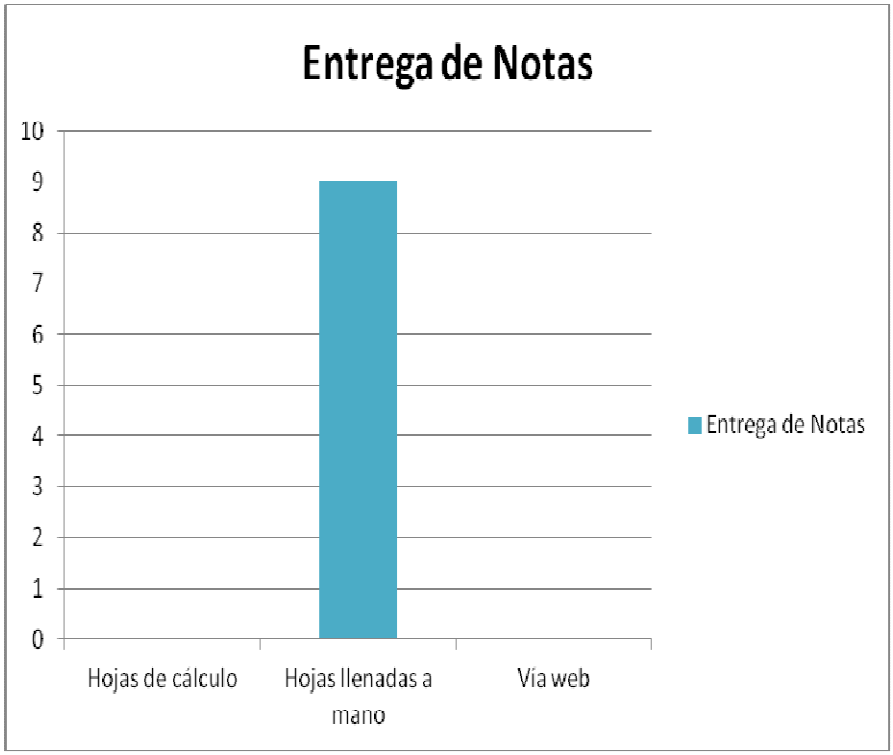



**Fig 3.5** Proceso de entregar de notas al personal administrativo

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

| 6.- ¿Cuál es el proceso que deben seguir al matricularse? | Proceso Matriculación |
|---|---|
| Por medio de un sistema académico | 0 |
| Formularios llenados a mano | 9 |

Tabla 3.6. Proceso de Matriculación

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

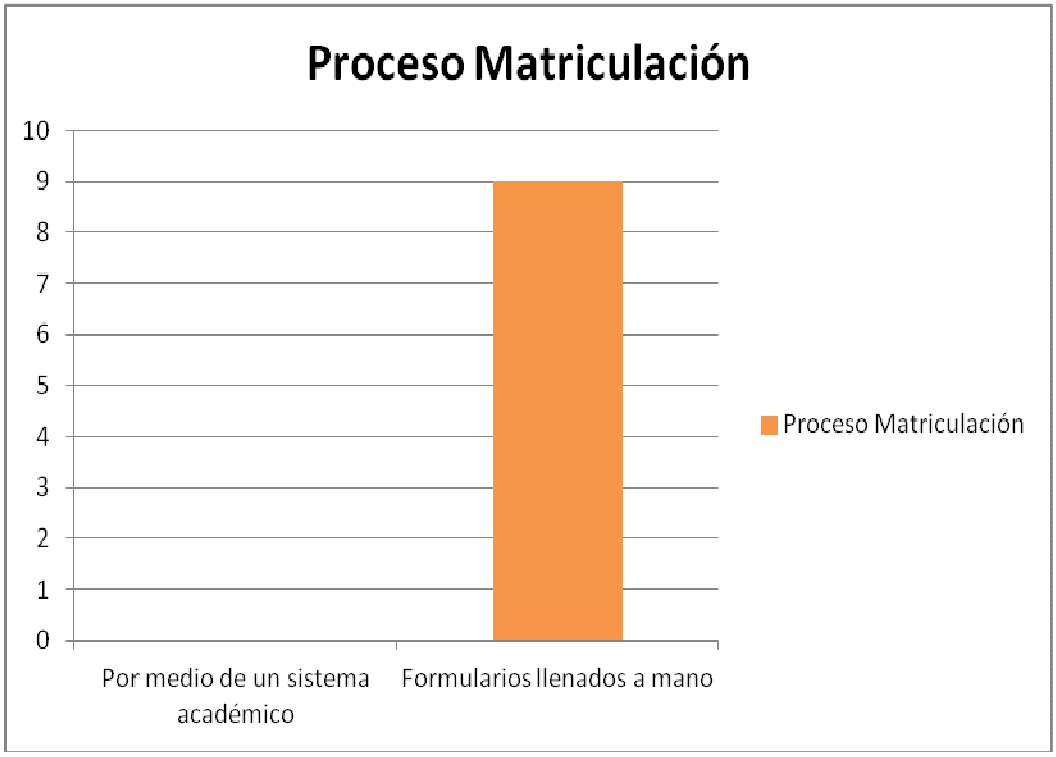



**Fig 3.6** Proceso de Matriculación

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

| 7.- ¿Cuál es el tiempo que se demora en matricular por alumno? | Tiempo de matriculación |
|---|---|
| **15 a 30 min** | 7 |
| **30 a 45 min** | 2 |
| **Más de 1 hora** | 0 |

Tabla 3.7. Tiempo para el proceso de matriculación

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

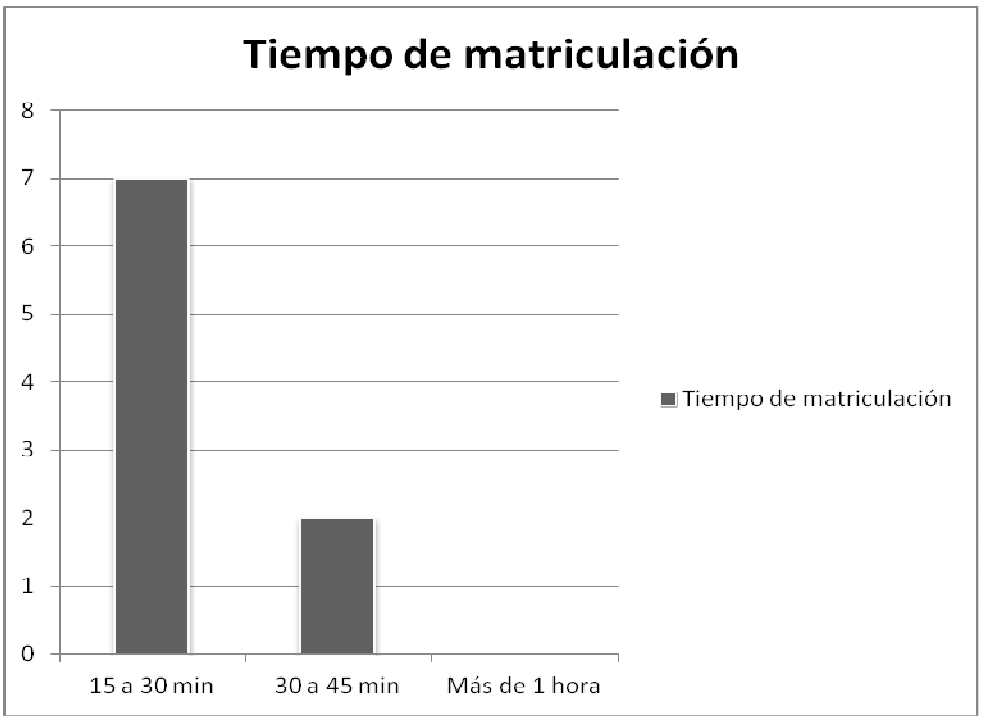



**Fig 3.7** Tiempo para el proceso de matriculación

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

En base al análisis de la encuesta realizada a los docentes de la Escuela Fiscal Mixta Ciudad de Ibarra se determinó lo siguiente:

1. Actualmente para el proceso de inscripción, el estudiante acude a la Institución con su representante, siempre y cuando sea para primero y segundo año de educación básica, para los otros años no se encuentra abierto este proceso. Se debe seguir los siguientes pasos para realizar la inscripción del estudiante:

   - Se solicita los siguientes requisitos para la inscripción del estudiante:

     Cédula del estudiante.

     Partida de nacimiento.

     1 foto tamaño carnet.

   - Se procede a llenar una ficha para la inscripción del alumno con los siguientes datos:

     Nombres y apellidos completos del estudiante.

     Fecha de nacimiento del estudiante.

     Lugar de nacimiento.

     Dirección de estudiante.

     Número de teléfono del estudiante.

     Nombre del representante.

     Dirección de representante.

Número de teléfono del representante.

Firma de representante.

- Una vez completa la ficha del estudiante se asigna una fecha para que rinda una prueba, si es aprobado se procede a la matriculación del estudiante.

2. El proceso de matriculación que realiza la Institución es de la siguiente manera:

- El estudiante de cualquier año de básica acude con su representante a la escuela en un periodo de fechas establecidas por el Ministerio de Educación, se le entrega un turno dependiendo de su llegada.

- El representante del estudiante debe asistir con la siguiente documentación para realizar el proceso de matriculación:

  Cédula del estudiante.

  Libreta del alumno del anterior año lectivo.

  1 foto tamaño carnet.

- Con los documentos antes descritos el docente que se encuentre designado para realizar el proceso de matriculación, recepta la

documentación y procede a llenar una ficha con los siguientes datos:

Nombres y apellidos completos del estudiante.

Fecha de nacimiento del estudiante.

Lugar de nacimiento.

Dirección de estudiante.

Número de teléfono del estudiante.

Nombre del representante.

Dirección de representante.

Número de teléfono del representante.

Firma de representante.

- Una vez completos los datos de la ficha del estudiante termina el proceso de matriculación.

3. Las calificaciones son llevadas manualmente por parte de los docentes los mismos que siguen el siguiente proceso:

   - Poseen una hoja por materia con los nombres de los estudiantes y columnas indicando el campo a la que pertenece la nota, por ejemplo Prueba 1.
   - Se les entrega una hoja trimestralmente para el registro de las notas durante el año lectivo y es guardada por cada docente de la manera que considere sea segura para su información.

4. Se tendrá un sistema con la posibilidad de alojarlo en la web pero por pedido de los docentes de la Escuela Fiscal Mixta Ciudad de Ibarra se montará en una intranet que le permita al docente de la Institución registrar sus notas a tiempo.

5. Se requiere automatizar el proceso de entrega de notas trimestralmente por medio de un sistema académico que entregue reportes de calificaciones acortando así los tiempos de entrega y ahorrando presupuesto para la compra de las respectivas libretas.

6. Se necesita agilitar los procesos administrativos como entrega de reportes para el personal docente de la Institución por medio de un sistema académico que facilite la entrega de los mismos.

7. Se requiere mejorar el tiempo para que los estudiantes realicen los procesos tanto de inscripción como de matriculación; llegando a la necesidad de realizar un sistema académico que posea un módulo dedicado a estos procesos, así mejorando los tiempos y calidad del servicio brindado al estudiante.

# CAPÍTULO IV

## 4. ETAPA DE DISEÑO

En esta etapa se traduce los requisitos en una representación que contenga todas las funciones y procesos necesarios para el desarrollo del sistema educativo, esto puede lograrse mediante varios métodos utilizados en la ingeniería de software como mapas de procesos, diagramas UML etc.

Toda la información obtenida en esta etapa debe ser muy clara ya que servirá como base fundamental al desarrollador del sistema en la etapa siguiente (codificación).

En la etapa de diseño del software se analizan distintas partes del programa:

- La arquitectura del software.
- La estructura de los datos.
- El detalle de los procesos.
- Las interfaces.

## 4.1 ARQUITECTURA N CAPAS

El modelo n-capas es una arquitectura predominante para la construcción de aplicaciones multiplataforma ya que permite tener desarrollos independientes, en cada capa facilita cambiar componentes y añadir nuevos módulos mejorando la funcionalidad, además que se puede manejar muchas peticiones con el mismo rendimiento.

### 4.1.1 CAPAS Y NIVELES

Dentro de la arquitectura del sistema (n-capas) encontramos 2 términos muy importantes los cuales debemos poder diferenciar ya que no significan lo mismo. El término capa se utiliza para referenciar a las distintas "partes" en que una aplicación se divide desde un punto de vista lógico (presentación, negocio, datos), mientras que "nivel" corresponde a la forma física en que se organiza una aplicación.

Debemos tener presente que las capas inferiores brindan servicios a las capas superiores.

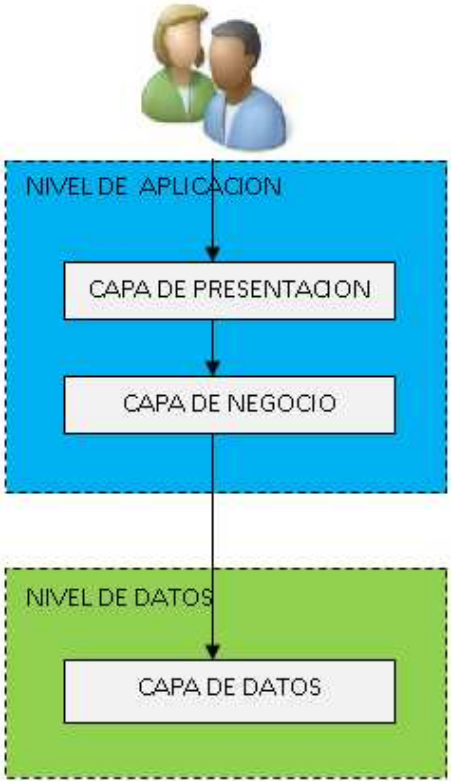



**Fig. 4.1** Capas y Niveles

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández.

Como se muestra en la Fig. 4.1 Capas y Niveles, presenta dos niveles, en el primero de ellos encontramos dos capas, de esta manera podemos decir que la capa de presentación interactúa con la capa de lógica de negocio, basándonos en una arquitectura en capas.

Como vimos en la Fig. 4.1 Capas y Niveles la arquitectura del sistema posee 3 capas (presentación, negocio, datos), y dos niveles (aplicación y datos) esto visto en la estructura del sistema académico sería de la siguiente manera:

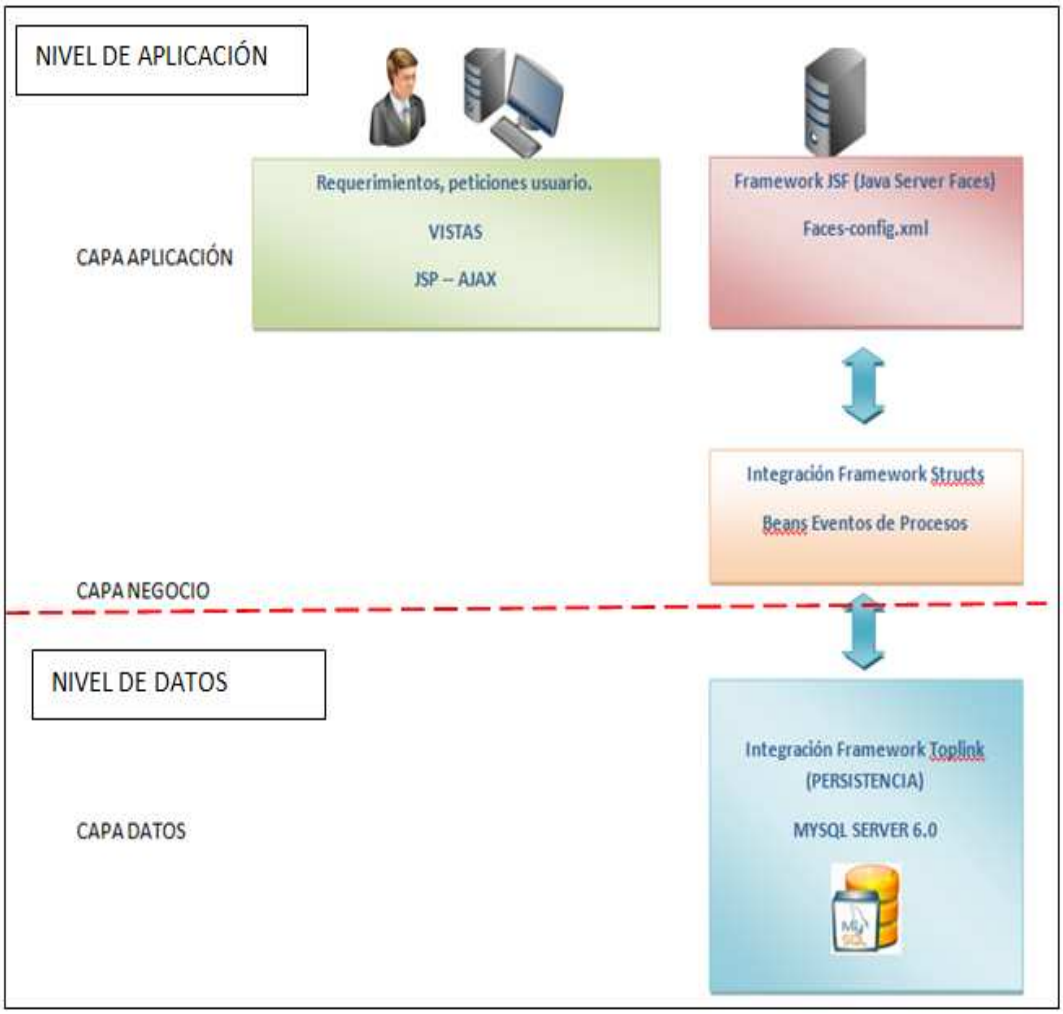



**Fig. 4.2** Capas y niveles del sistema

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández.

### 4.1.2 ¿POR QUÉ ARQUITECTURA DE 3 CAPAS?

Se implementó para el sistema una arquitectura en tres capas, puesto que supera a otras arquitecturas, mejorando el rendimiento y proporcionando mayor flexibilidad, fácil reutilización y escalabilidad.

La razón primordial por la que se aplicó esta arquitectura es la separación del sistema en capas, de esta manera se logra independencia entre cada una de ellas, por lo que el desarrollo se facilita, ya que los aspectos importantes estarán encapsulados según las acciones que realice.

Capa presentación = Interfaz gráfica.

Lógica de negocio = Procesos.

Capa datos = Datos de entrada y salida.

Estas capas serán independientes unas de las otras.

#### 4.1.2.1 CAPA DE PRESENTACIÓN

En la capa de presentación encontraremos toda la parte visual, es la parte con la que el usuario va a interactuar y a la cual se le va a hacer las peticiones, por lo que debe ser la parte más llamativa del sistema, muy descriptiva para que no presente mayor complejidad.

En esta capa se encontraran herramientas y tecnologías que nos permitirán presentar al usuario una vista mucho más ágil y agradable para su uso.

La capa de presentación está localizada en el nivel de aplicación en donde encontramos dos capas:

- La capa de presentación en la cual están los *frameworks* que nos permiten crear la interfaz de usuarios, JSF 1.2, AJAX (*ajax4jsf-1.1.1.jar*).

- La capa de negocio en donde podemos hablar del *Netbeens 6.7* como herramienta de desarrollo y en la cual encontraremos todo el negocio, es decir los procesos.

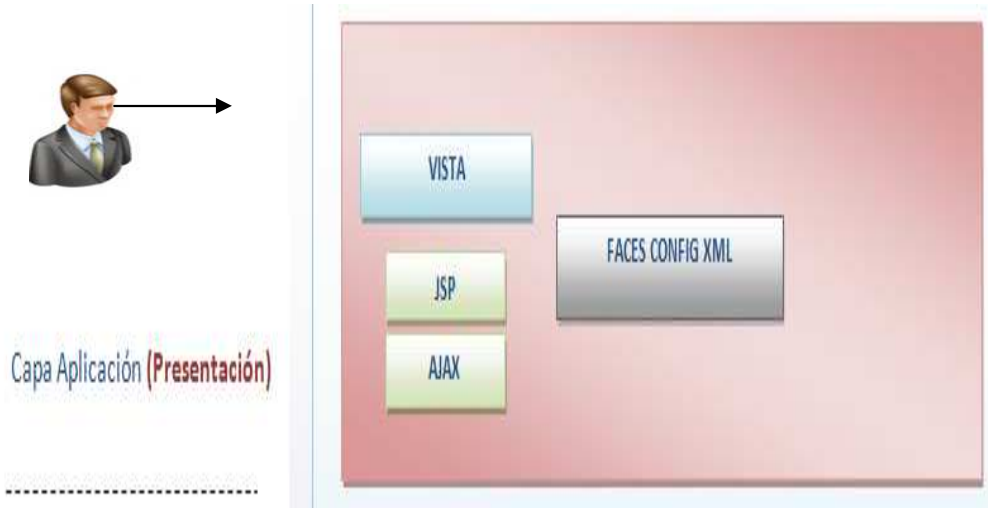



**Fig. 4.3** Capa Presentación

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández.

En la capa de negocio se utiliza la tecnología *Java Server Faces* (JSF) versión 1.2 es un *framework* para aplicaciones web basadas en tecnología JAVA que permite construir una biblioteca de componentes reutilizables, personalizados, ayuda a encapsular el interior de un componente, permitiendo ocultar el código complejo a los usuarios finales sin embargo, del que lanza muchas peticiones al servidor pero al complementarse con AJAX; una tecnología que permite hacer una petición a una URL sin refrescar la página actual del navegador.

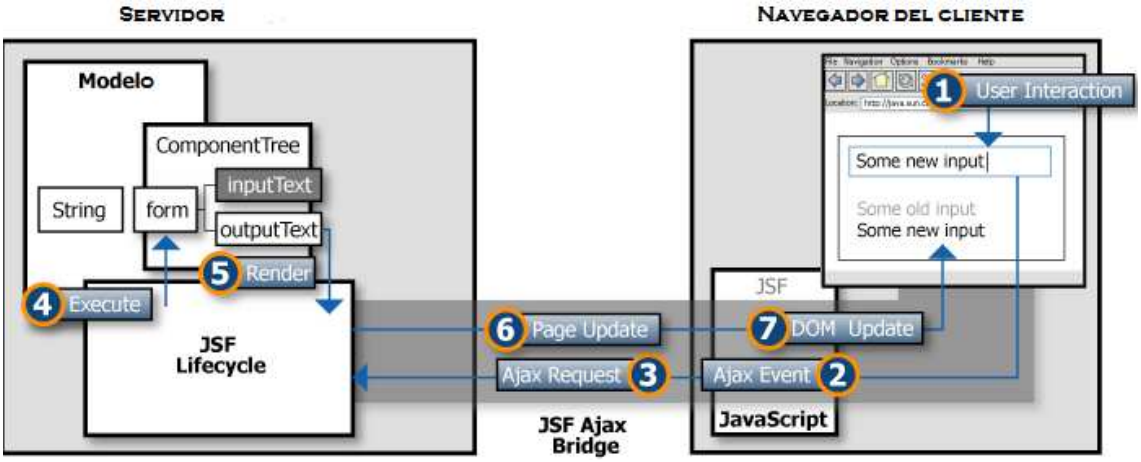



**Fig. 4.4** Funcionamiento JSF, AJAX

FUENTE: wiki.icesoft.org/display/ICE/Automatic+Ajax

Según los pasos de la Fig. 4.4 el proceso se describiría de la siguiente manera:

En el paso 1 el usuario interactúa con el sistema y realiza una petición.

En el paso 2 la solicitud es escuchada por el AJAX que se encuentra referenciado en el JSP.

En los pasos 3 y 4 la solicitud procesada va al servidor con la información suficiente para que el respectivo *Bean* la procese.

Y por último en los pasos 5, 6 y 7 la respuesta sale del servidor y actualiza la página indicada con su respectivo estilo.

#### 4.1.2.2 CAPA DE NEGOCIO

La capa de negocio es donde residen los programas que se ejecutan, llegan las peticiones del usuario desde la capa de presentación y se envían las respuestas una vez realizado el proceso correspondiente. Es en esta capa es donde se establecen todas las reglas que deben cumplirse.

Esta capa también se comunica con la capa de datos para solicitar al gestor de base de datos almacenar o recuperar datos de las tablas.

Esta capa interactúa con la capa de capa de presentación, lo que significa que la capa de lógica de negocios presenta una "interfaz" para brindar servicios a la capa de presentación.


**Fig. 4.5** Capa Presentación

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández.

#### 4.1.2.3 CAPA DE DATOS


Se trata generalmente de un motor de base de datos que puede o no ejecutarse en el mismo equipo. Esta capa brinda servicios a las capas superiores. Aquí llegan las peticiones desde la capa de negocio de esta manera se insertan o se devuelven datos.

##### 4.1.2.3.1 BASE DE DATOS

El motor de base de datos utilizado en el sistema es *MySQL 6.0*, puesto que presta características como multiusuario, multiplataforma y de código abierto; convirtiéndose así en el motor de base de datos más popular del mundo. Dicho motor o gestor de datos es generalmente utilizado en aplicaciones web por su estabilidad, fácil manejo e interpretación del lenguaje SQL para consultas.

##### 4.1.2.3.2 CARACTERÍSTICAS GENERALES

- Base de datos relacional.
- Utiliza múltiples tablas para almacenar y organizar la información.
- Multihilo porque realiza varias tareas recurrentemente.
- Multiusuario pudiendo conectarse varios usuarios simultáneamente.

- Multiplataforma puesto que es adaptable a la mayoría de SO.

#### 4.1.2.3.3 VENTAJAS

- Es una base de datos muy rápida en la lectura de aplicaciones web, convirtiéndola en la herramienta ideal para este tipo de aplicaciones.

- Además está disponible en gran cantidad de plataformas y sistemas, brindando a su vez, una conectividad muy segura.

- Nos da también una potencia sin igual, ya que utiliza SQL como lenguaje para consultar la base de datos, dicho motor optimiza enormemente gran cantidad de trabajo.

- Es portable por la misma característica anterior, SQL es fácilmente portable a otras plataformas y nos brinda una gran escalabilidad, pudiendo manipular base de datos enormes con más de 50 millones de registros.

### 4.1.2.4 CONEXIÓN CON LA APLICACIÓN

Para la conexión de la base de datos con la aplicación es necesario el *driver mysql-connector-java-5.1.7-bin* que es importado en *Netbeans 6.7* el cual

genera los *strings* de conexión, llamados o incluidos como librerías dentro de la aplicación.

En el caso del sistema académico solo generaremos una conexión con esta base de datos, motivo por el cual se puede establecer una parametrización por propiedades de la herramienta de desarrollo, de esta manera configuramos el nombre de la base de datos, puerto predeterminado por lo general 3306, usuario y contraseña.

Como habíamos mencionado anteriormente la unidad de persistencia está definida por un contenedor extensión .XML de archivo llamado *persistence.xml*; en el cual parametrizamos inicialmente con *framework* ORM a la base de datos.

#### 4.1.2.5 DISEÑO DEL MODELO DE LA BASE DE DATOS

Para el diseño de la base de datos se analizaron los requerimientos de los usuarios, para que así cada una de las tablas contenga la información necesaria para la aplicación.

Teniendo el conocimiento de lo que se necesita con respectos a datos se utilizó técnicas que nos permiten el “modelado” de la base de datos.

Como se ha mencionado antes el diseño de la base de datos tiene la misma importancia que el de desarrollo de la aplicación, por lo que al igual que esta se debe seguir un proceso ordenado de pasos que aseguren un diseño eficaz.

Es muy importante seguir todos los pasos del diseño, esto ahorrara mucho tiempo más adelante. Sobre todo si alguna vez se tiene que modificar la base de datos para corregir errores o para implementar alguna característica nueva, algo que sucede con mucha frecuencia en las aplicaciones.

La primera fase del diseño de una aplicación o una base de datos consiste en hablar con el cliente para conocer sus requerimientos y que necesitan ellos de la aplicación.

Una vez establecidas las necesidades, se inicia el diseño tomando en cuenta los 3 siguientes pasos fundamentales para a lograr el objetivo, una base de datos útil que cumpla con los requisitos y que durante el desarrollo o en futuras modificaciones no sea necesario afectar radicalmente a su diseño por mínimos cambios.

Los tres siguientes pasos son la generación de tres modelos:

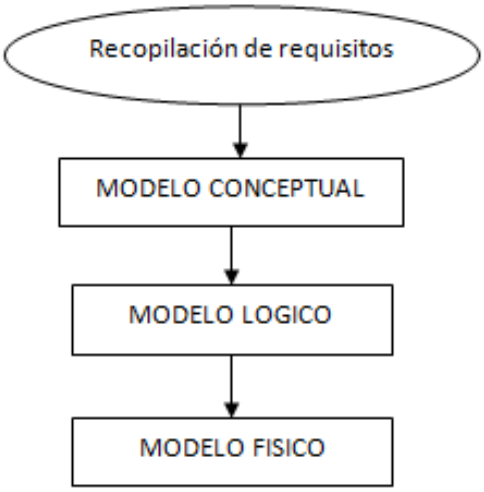



**Fig. 4.6** Modelos de la base de datos

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández.

El primer paso a seguir es la elaboración de los diagramas conceptuales los cuales no están orientados a ningún sistema físico concreto o SGBD (Sistema de gestión de base de datos), y no tienen una orientación informática clara por lo que puede describir rápidamente ya que por su simplicidad no necesita mayor investigación, Así que el modelo conceptual es una representación gráfica que nos facilita una forma clara de ver el proceso en su totalidad, por lo que podrían servir para explicar fácilmente a un usuario cómo funciona el proceso de una forma grafica.

El siguiente paso es convertir el modelo conceptual en un modelo lógico.

#### 4.1.2.5.1 MODELO LÓGICO

El modelo lógico permite tener una visión más clara de la estructura, flujo y relaciones de la base de datos.

El modelo lógico utilizado para la base de datos es el modelo relacional; este modelo describe una base de datos de forma sencilla y global ya que representa un esquema con entidades y asociaciones.

En este modelo se observa las estructuras de cada una de las tablas, relaciones, tipos de relaciones y flujo de datos.

El objetivo de este modelo es obtener una representación que se use de la manera más eficiente los recursos disponibles al momento de estructurar datos, modelar restricciones y relaciones.

Para obtener este modelo se utilizó el gestor de base de datos MSQL, como se muestra en la Fig. 4.7. Modelo Lógico

**Fig. 4.7** Modelo Lógico

#### 4.1.2.5.2 MODELO FÍSICO

El siguiente paso es obtener un modelo físico a partir del modelo lógico antes visto. Para esto se deben tener claras las estructuras de almacenamiento y métodos usados para tener un acceso efectivo a los datos.

El diseño físico se adaptara al SGBD (Sistema gestor de base de datos) específico que se va a utilizar (MSQL).

Como muestra la Fig. 4.8 el Modelo Físico de la base de datos.

**Fig. 4.8 Modelo Físico**

#### 4.1.2.6 DICCIONARIO DE DATOS

"Un diccionario de datos es un conjunto de metadatos que contiene las características lógicas y puntuales de los datos que se van a utilizar en el sistema que se programa, incluyendo nombre, descripción, alias, contenido y organización"[5].

**Véase anexo 2.**

### 4.1.3 DISEÑO UML

En base a los diagramas de caso de uso se diseña la aplicación puesto que estos describen comportamiento del sistema, por lo tanto los diagramas de caso de uso indican el funcionamiento del negocio. Como se muestra a continuación con los siguientes diagramas de caso de uso.

---

[5] Referencia tomada http://es.wikipedia.org/wiki/Diccionario_de_datos, línea 1 y 2.

**Diagrama de casos de usos general con Perfil Administrador**

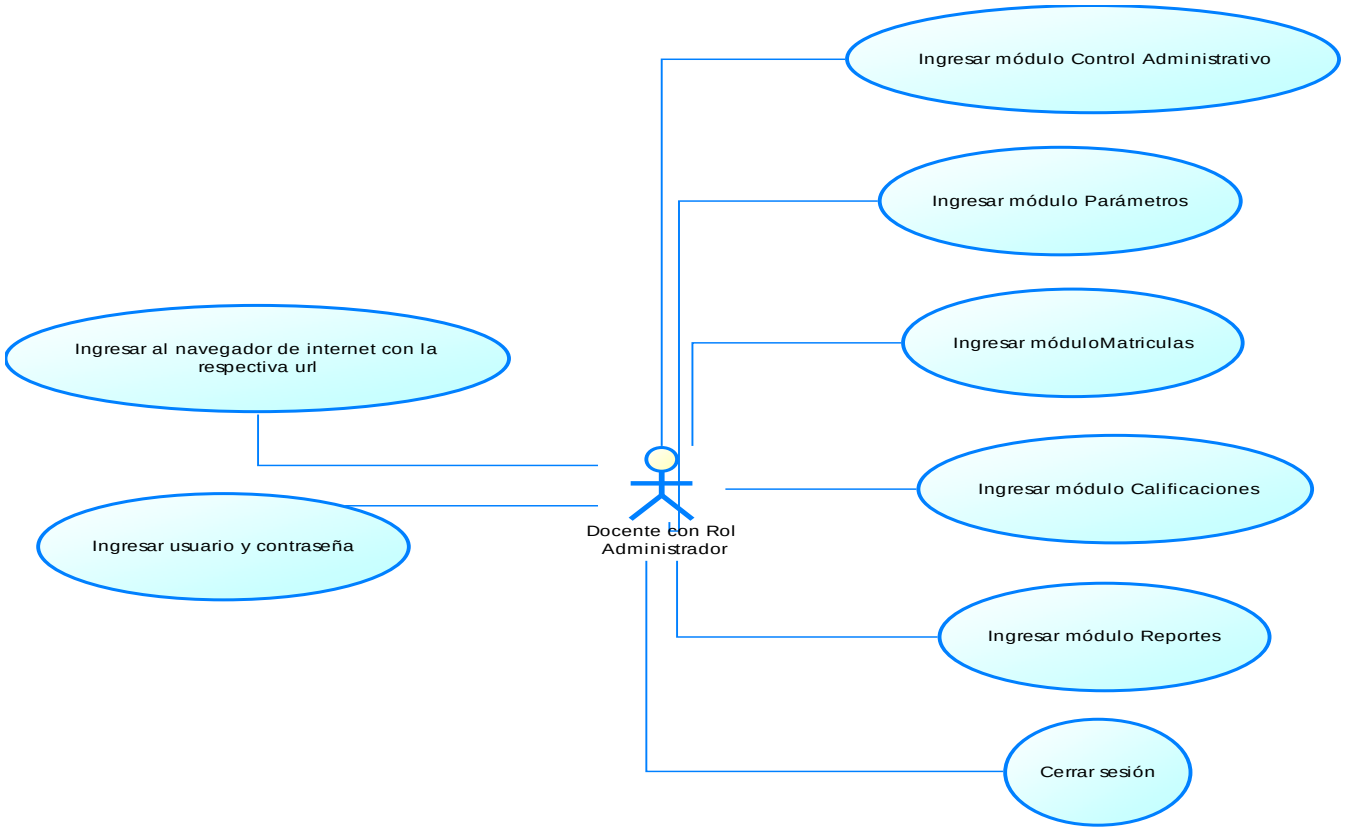



**Fig. 4.9** Casos de usos general con Perfil Administrador

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo Control Administrativo con perfil Administrador**

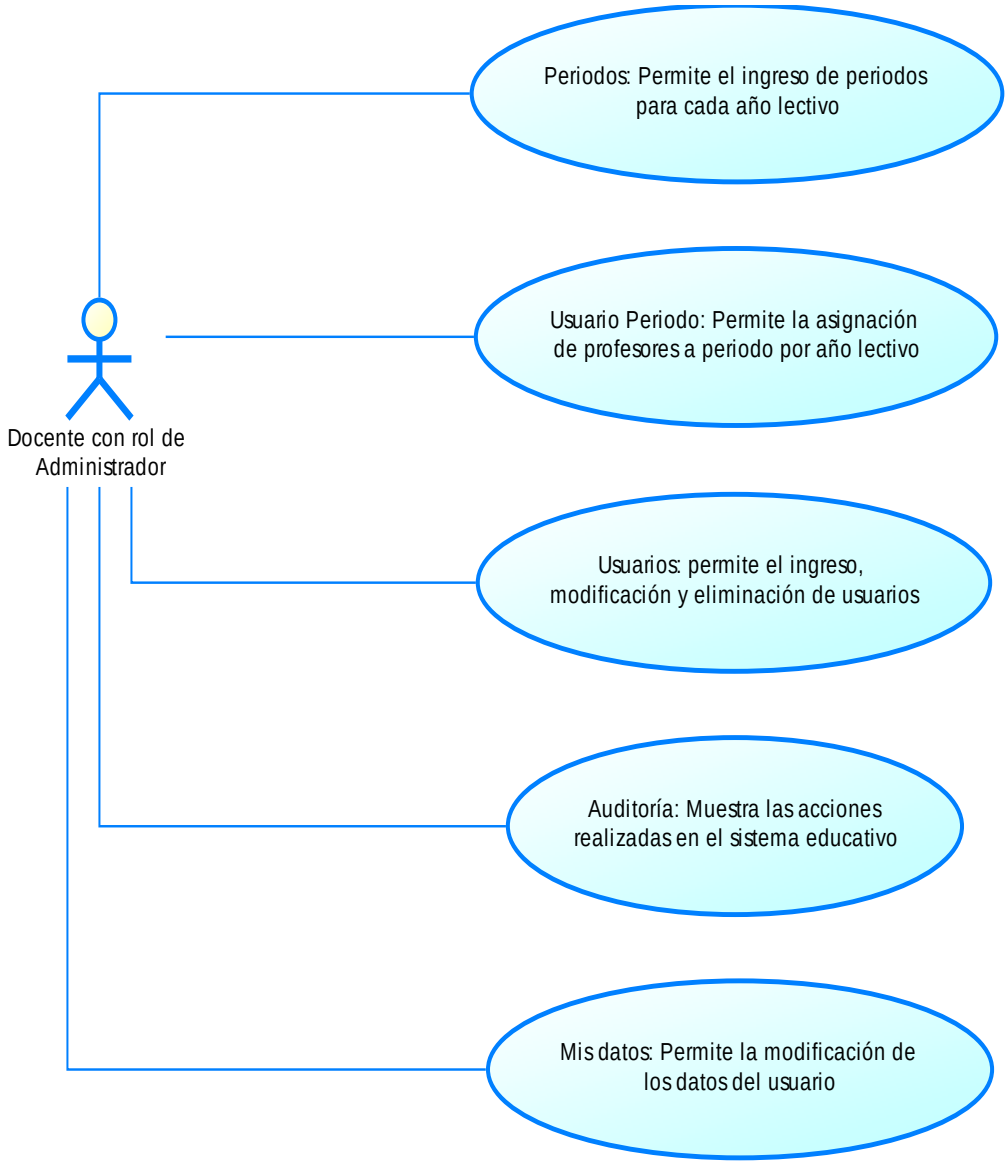



**Fig. 4.10** Casos Módulo Control Administrativo con perfil Administrador

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo parámetros con Perfil Administrativo**

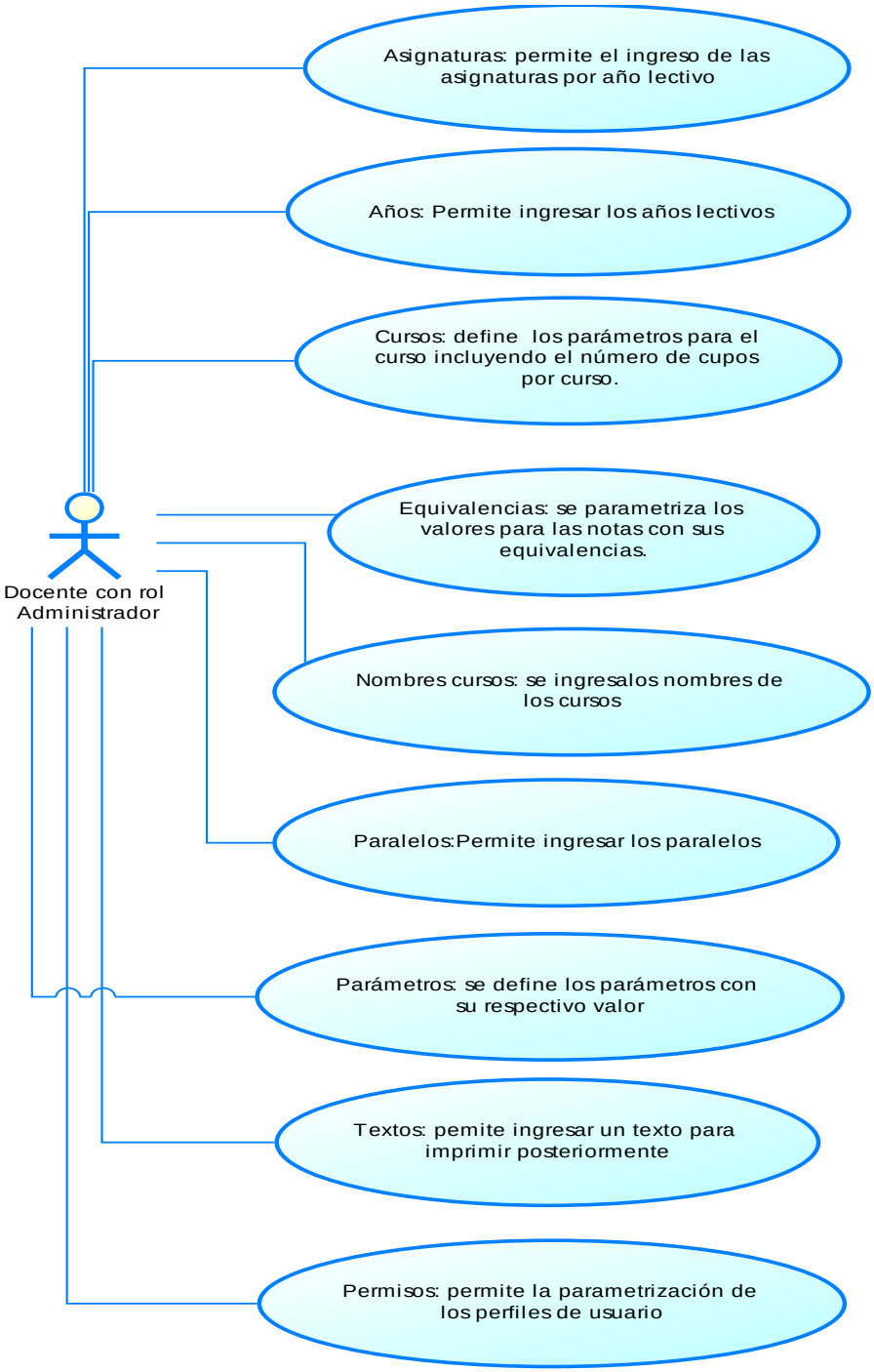



**Fig. 4.11** Diagrama de casos Módulo parámetros con perfil Administrativo

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo Matrículas con Perfil Administrativo**

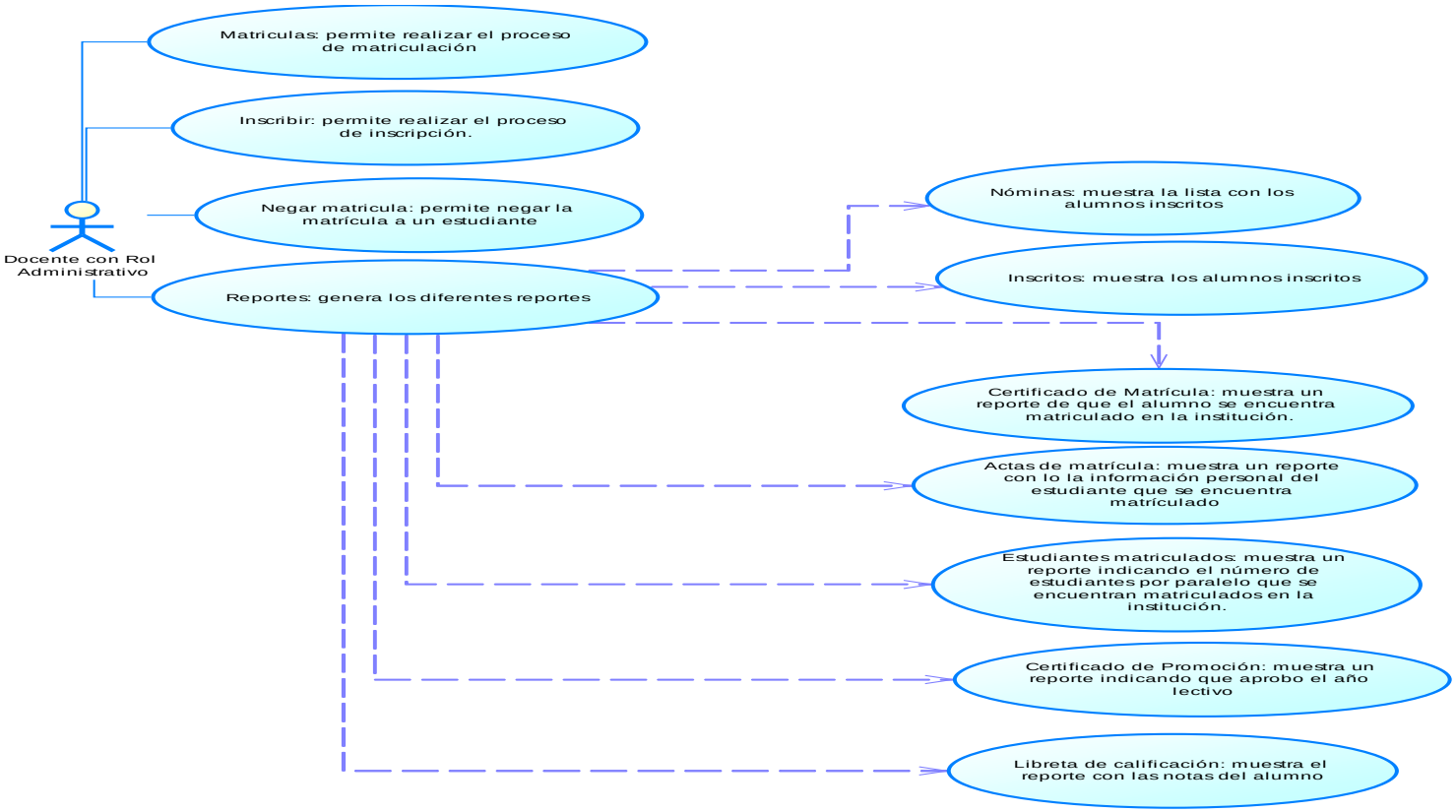



**Fig. 4.12** Casos Módulo Matrículas con Perfil Administrativo

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo de Calificaciones con Perfil Administrativo**

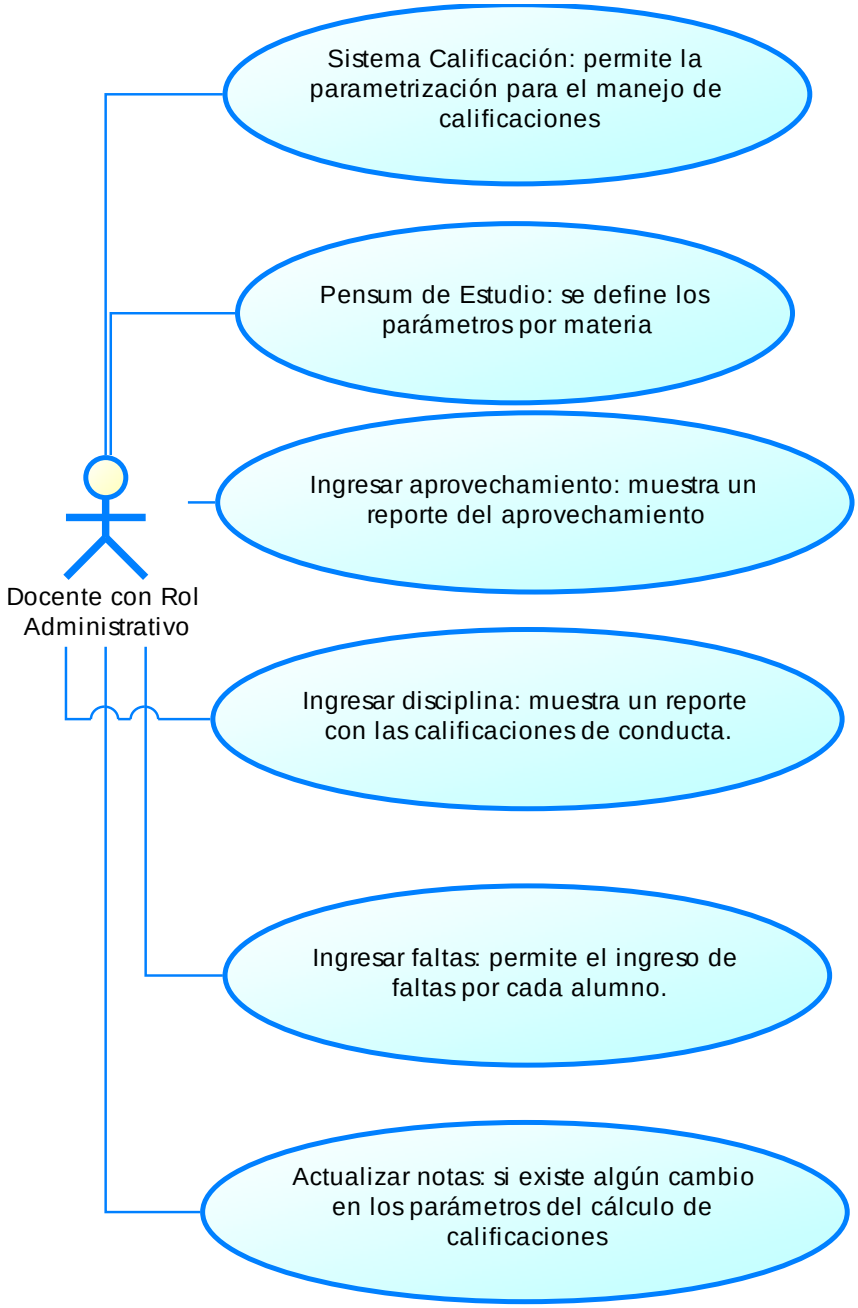



**Fig. 4.13** Casos Módulo de Calificaciones con Perfil Administrativo

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo reportes con Perfil Administrativo**

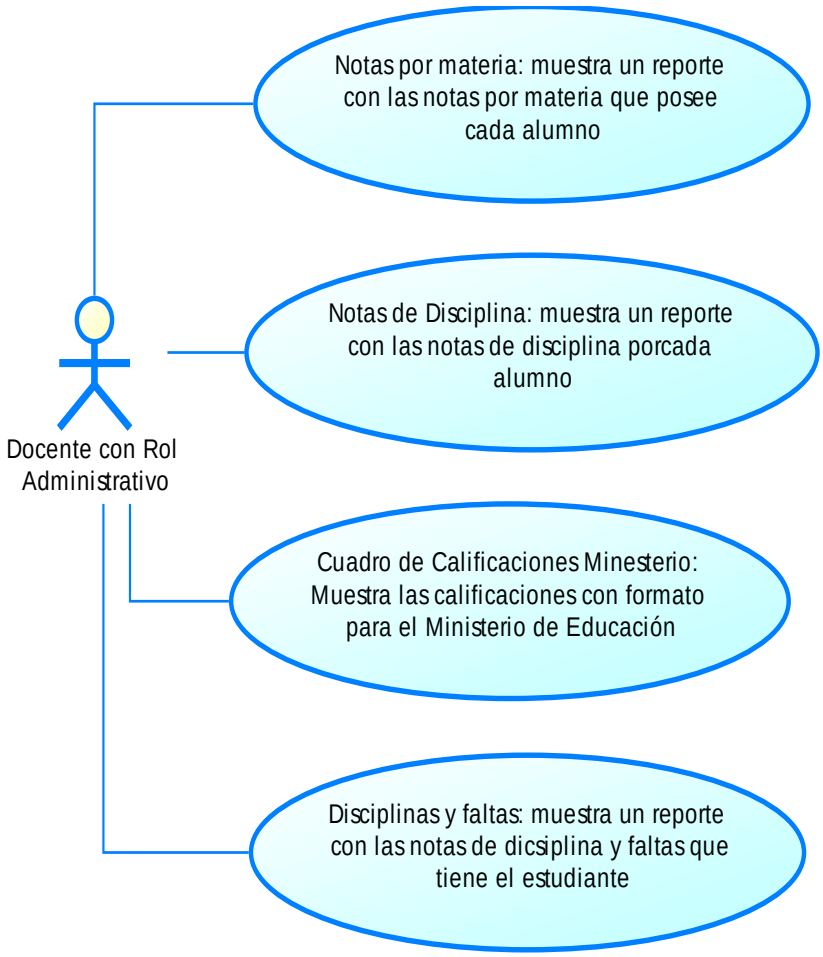



**Fig. 4.14** Casos Módulo reportes con Perfil Administrativo

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Diagrama de casos de usos general con Perfil Profesor**

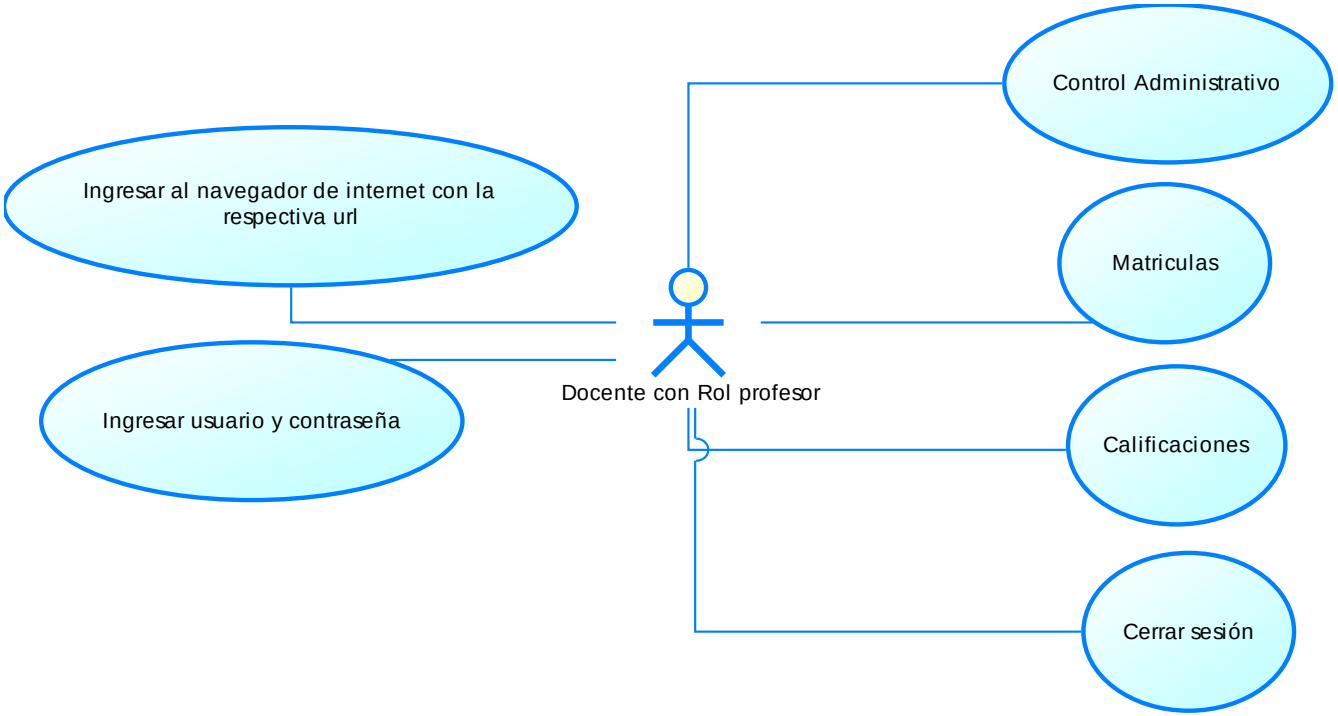



**Fig. 4.15** Casos de usos general con Perfil Profesor

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo Control Administrativo con perfil de Profesor**

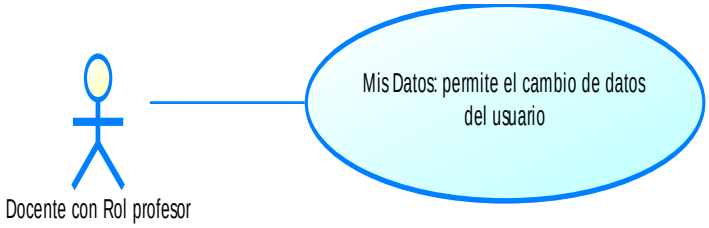



**Fig. 4.16** Casos de usos general con Perfil Profesor

FUENTE: Tesista(s)

AUTOR: Andrea Mora, María Elisa Hernández

**Módulo Matrículas con Perfil de Profesor**

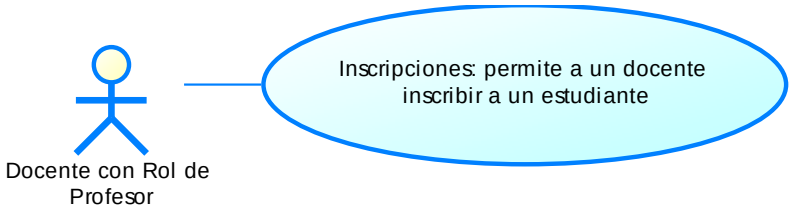



**Fig. 4.17** Casos de usos Módulo Matrículas con Perfil de Profesor

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo de Calificaciones con Perfil de profesor**

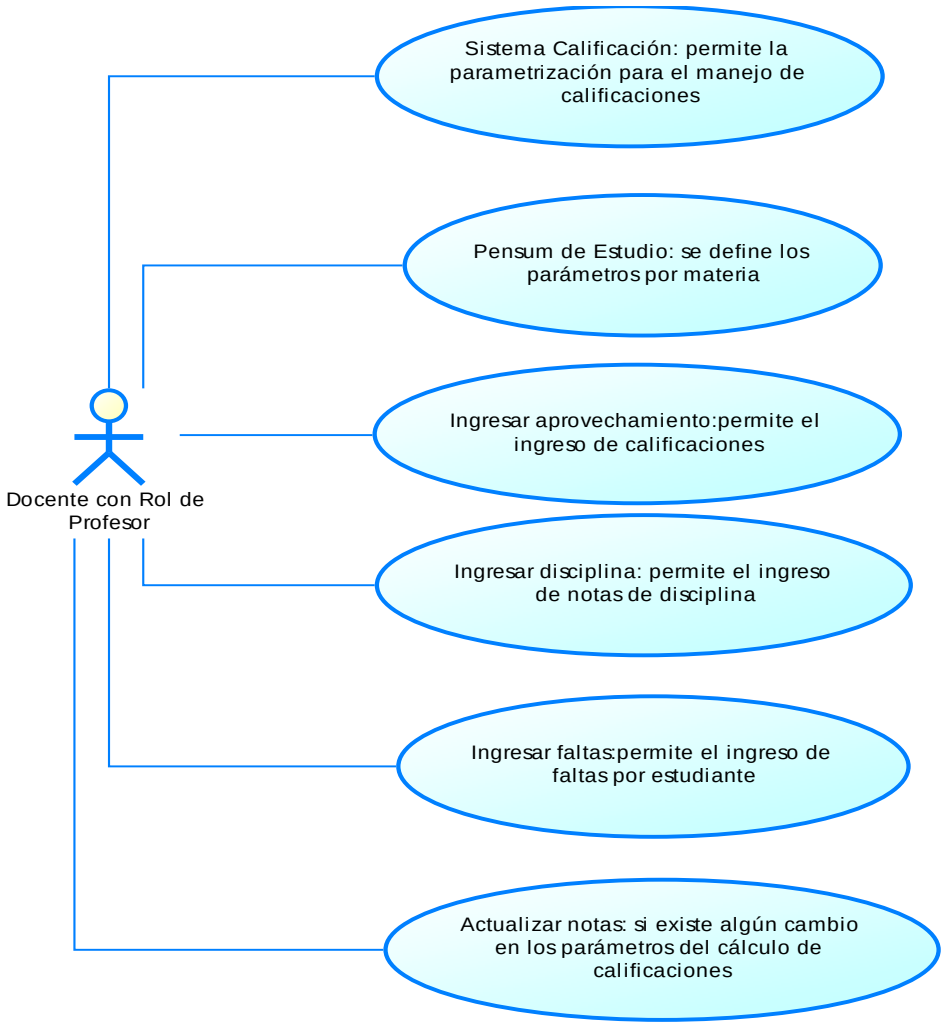



**Fig. 4.18** Casos de usos Módulo de Calificaciones con Perfil de profesor

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Diagrama de casos de usos general con Perfil Secretaria**

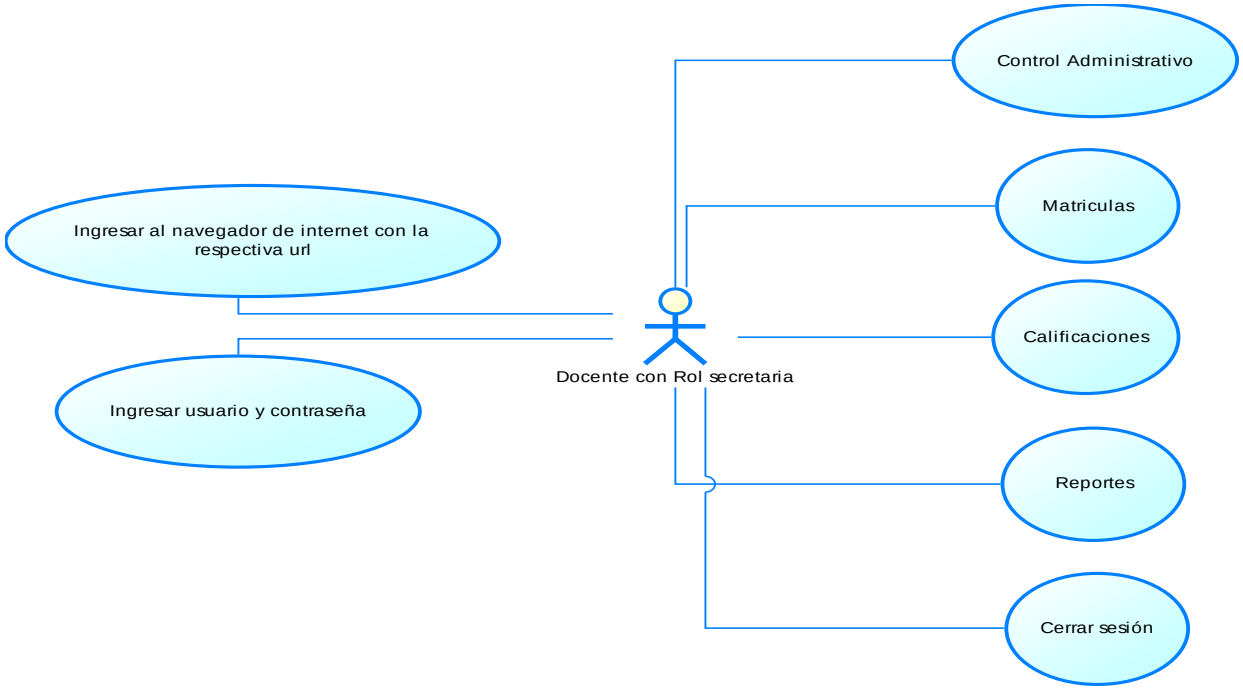



**Fig. 4.19** Casos de usos general con Perfil Secretaria

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo Control Administrativo con perfil de Secretaria**

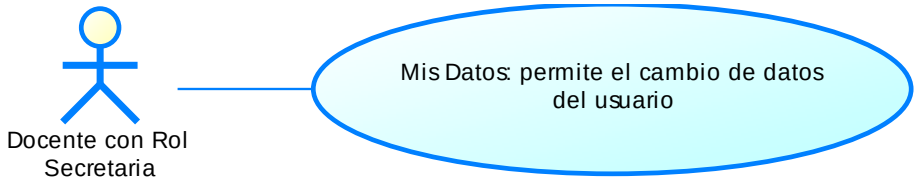



**Fig. 4.20** Casos de usos Módulo Control Administrativo con perfil de Secretaria

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

## Módulo Matrículas con Perfil Secretaria

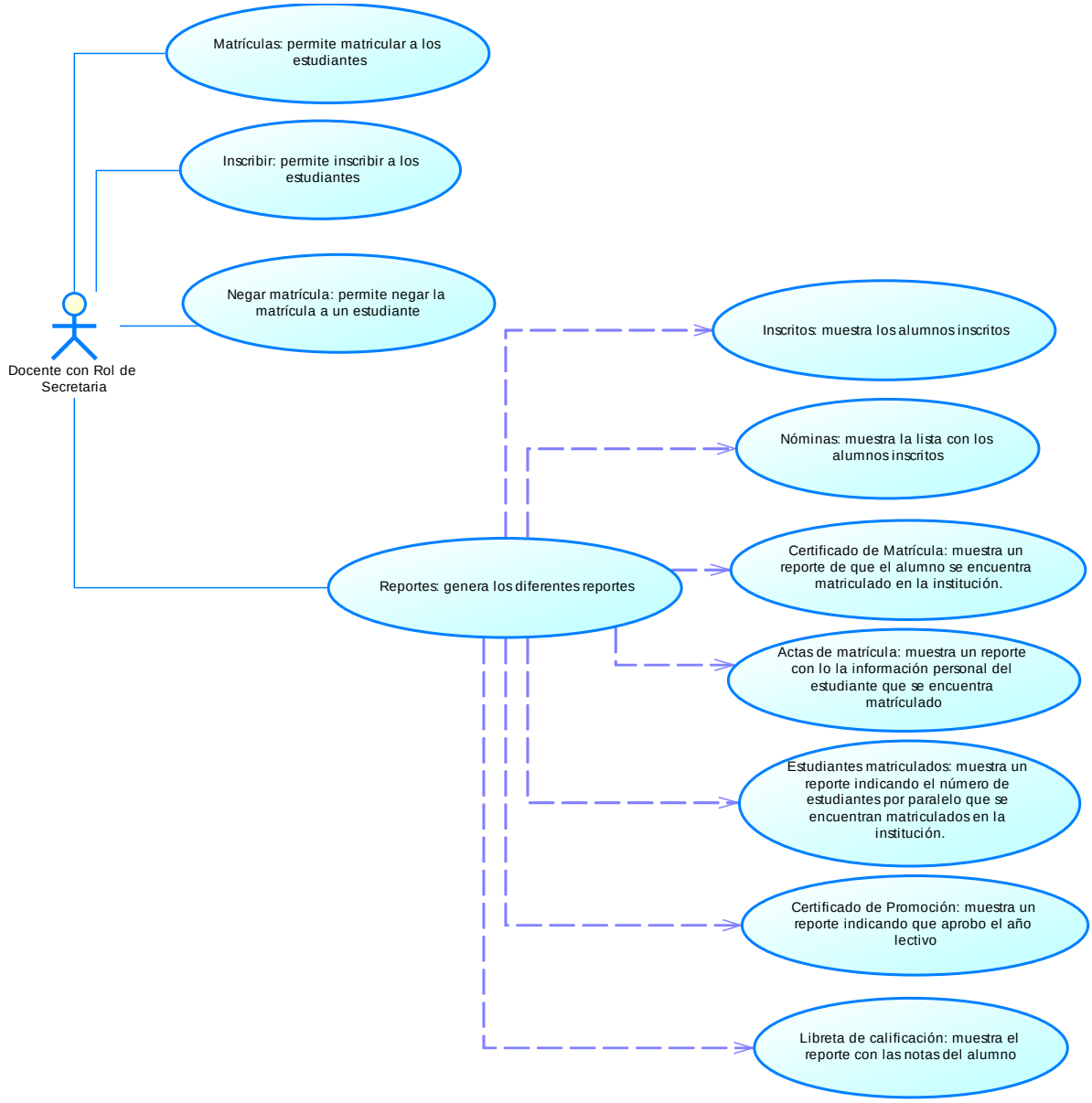



**Fig. 4.21** Casos de usos Módulo Matrículas con Perfil Secretaria

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo de Calificaciones con Perfil de Secretaria**

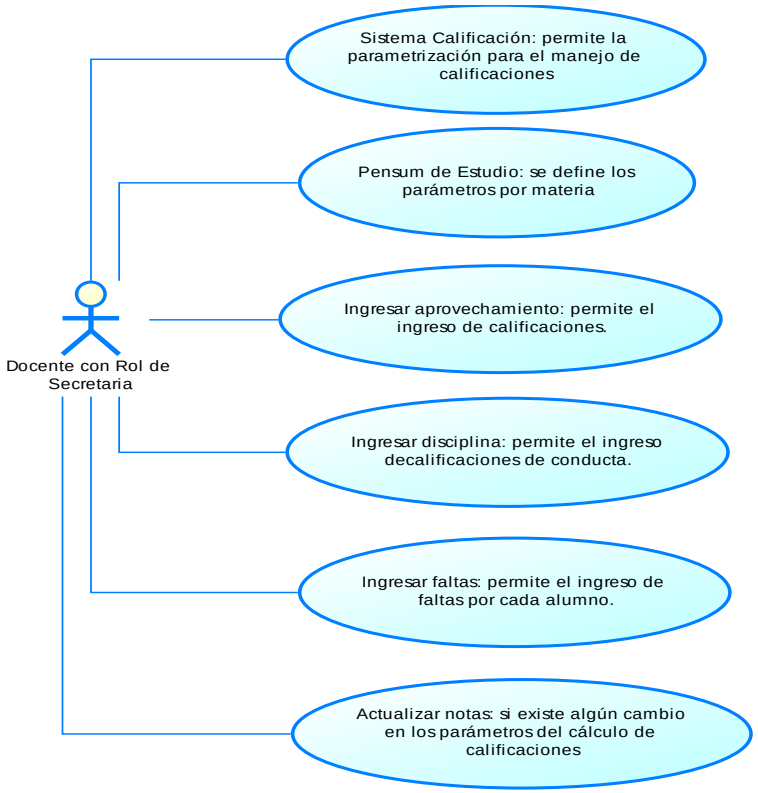



**Fig. 4.22** Casos de usos Módulo de Calificaciones con Perfil de Secretaria

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Diagrama de casos de usos general con Perfil Inspector**

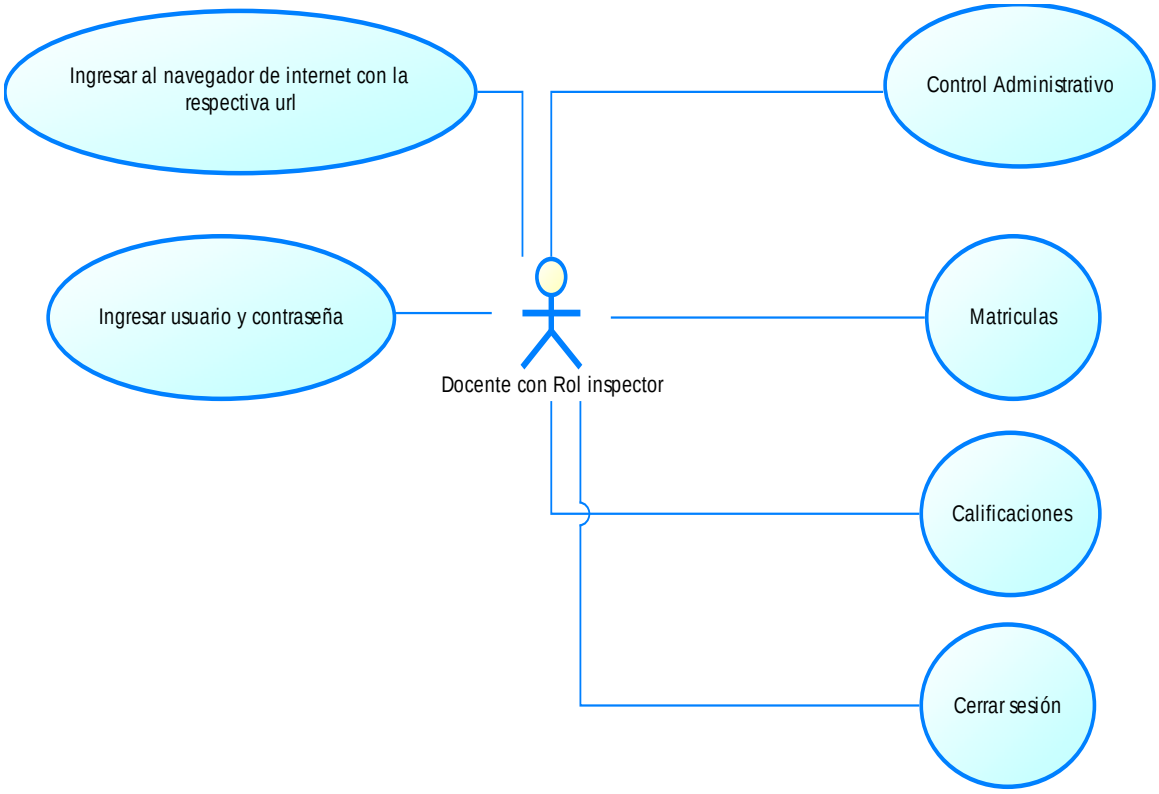



**Fig. 4.23** Casos de usos general con Perfil Inspector

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo Control Administrativo con perfil Inspector**


**Fig. 4.24** Casos de usos Módulo Control Administrativo con perfil Inspector

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo Matrículas con Perfil Inspector**

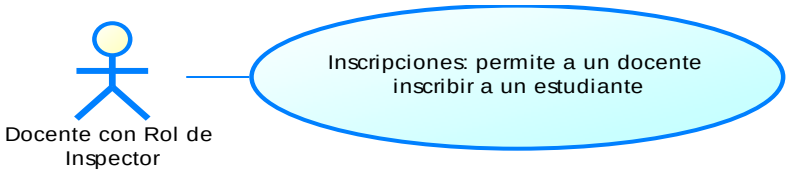



**Fig. 4.25** Casos de usos Módulo Matrículas con Perfil Inspector

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Módulo de Calificaciones con Perfil de Inspector**

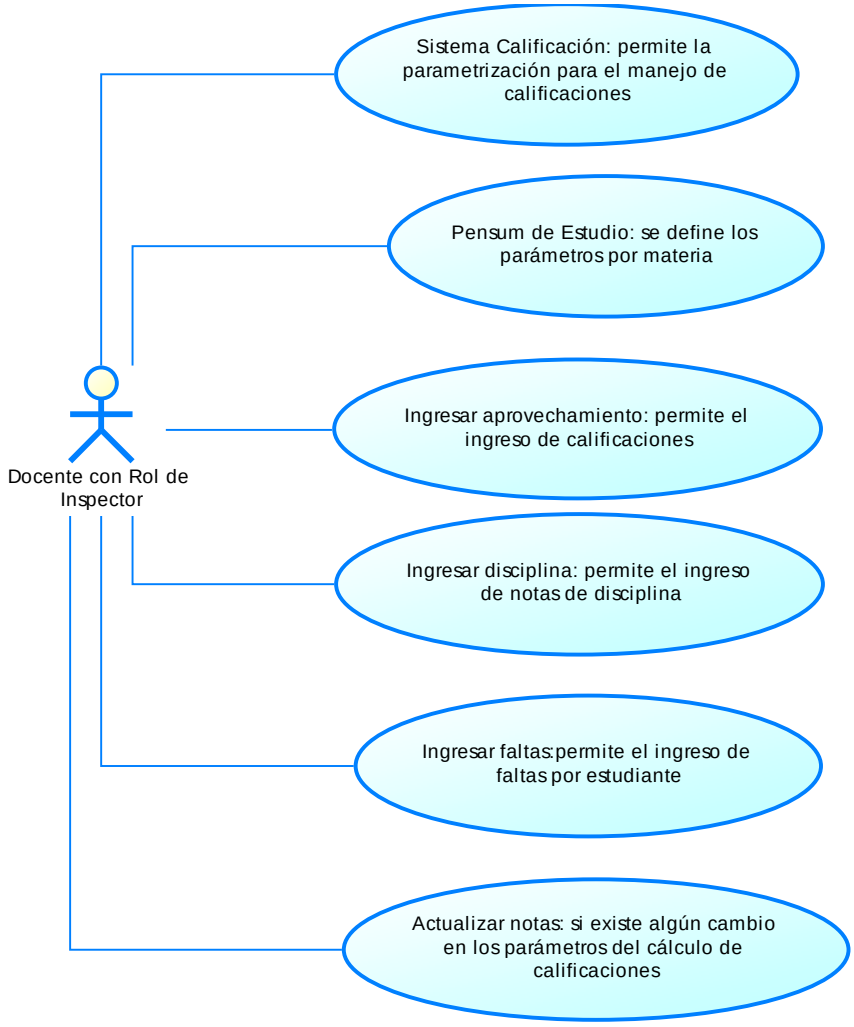



**Fig. 4.26** Casos de usos Módulo de Calificaciones con Perfil de Inspector

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

### 4.1.4 IDENTIFICACIÓN DE SUCESOS

Los sucesos representa la comunicación entre los docentes con el sistema, con el fin de identificar los sucesos.

La identificación de sucesos es un diagrama que muestra cada objeto como una línea vertical y cada suceso como una línea horizontal que va desde el emisor al receptor. El tiempo aumenta desde arriba hacia abajo y muestra la secuencia de los sucesos más no la temporización exacta.

Fig. 4. 25 representa una traza de sucesos para el ingreso de usuarios al sistema educativo.

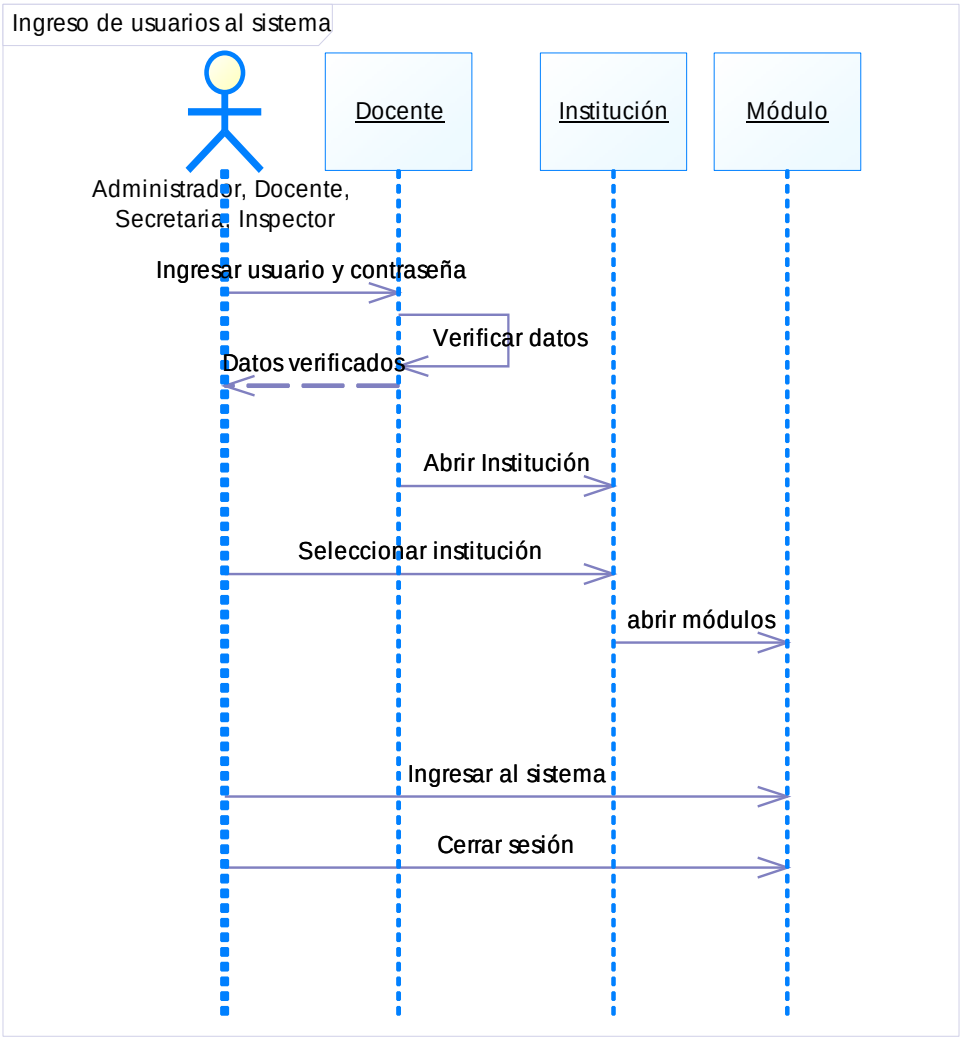



**Fig. 4.27** Seguimiento de sucesos para el ingreso de usuarios al sistema

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

Fig. 4.26 representa una traza de sucesos para el proceso de inscripción de un estudiante.

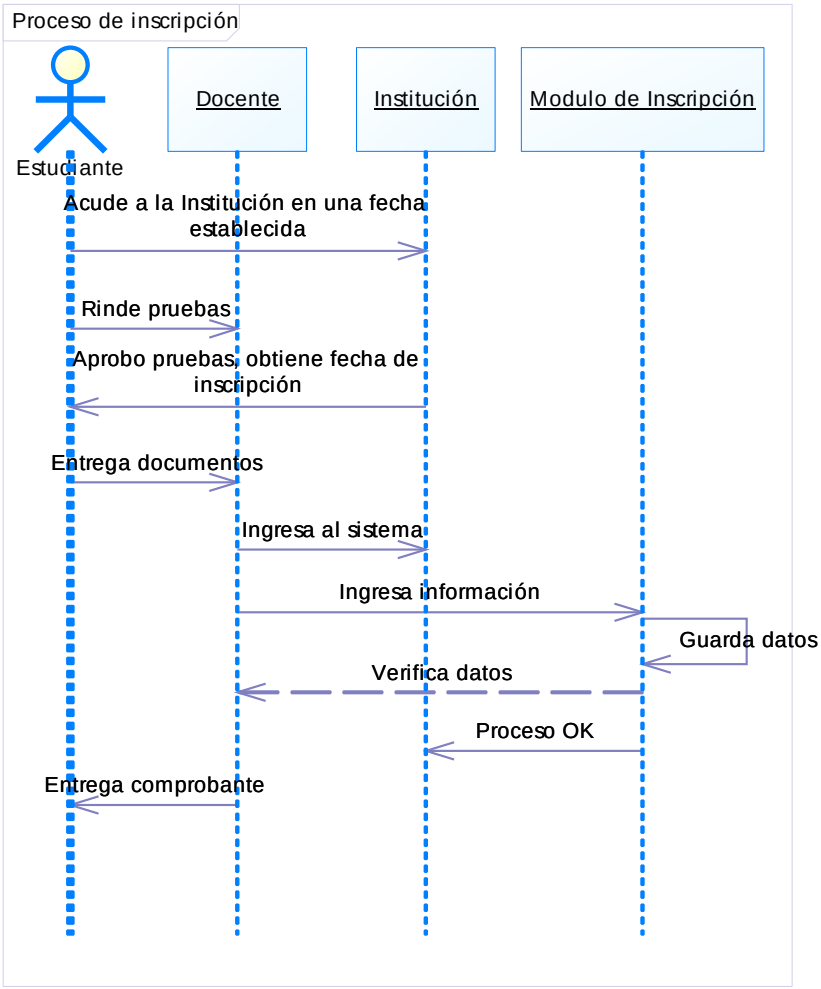



**Fig. 4.28** Seguimiento de sucesos para el proceso de inscripción

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

Fig. 4.27 representa una traza de sucesos para el proceso de matriculación de un estudiante.

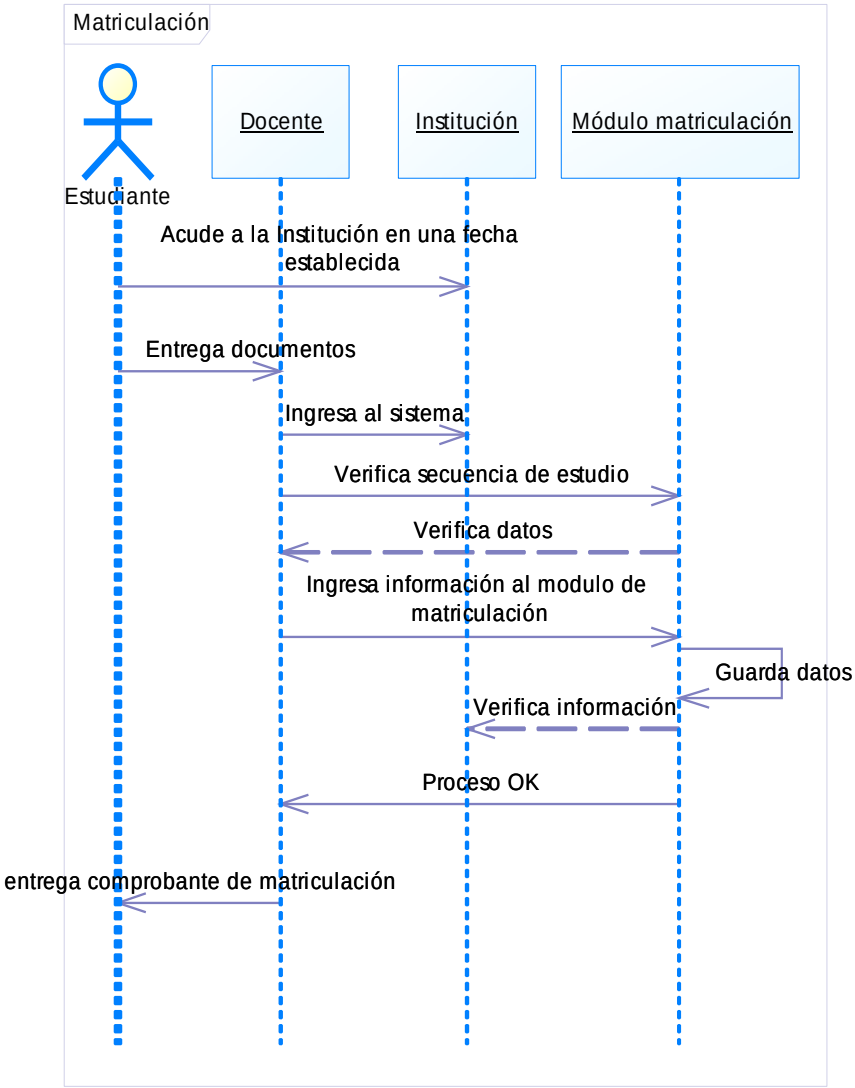



**Fig. 4.29** Seguimiento de sucesos para el proceso de matriculación

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

### 4.1.5 DISEÑO DE DIAGRAMA DE CLASES

Un diagrama de clases es el principal para el análisis y diseño de una aplicación ya que describe la estructura del sistema mostrando sus clases, atributos y relaciones entre ellos, donde:

**Atributo:** son características, valores que corresponden a un objeto.

**Clases:** es la unidad básica que encapsula los datos de un objeto.

**Relaciones:** es la manera con la que se pueden comunicar 2 o más clases, existen diferente tipo de relaciones como: composición, agregación, asociación, etc.

La Fig. 4.28 muestra el Diagrama de paquetes de clases para el diseño de la aplicación.

El resto de figuras que se muestran a continuación del Diagrama de paquetes son los diagramas de clases que contiene cada paquete.

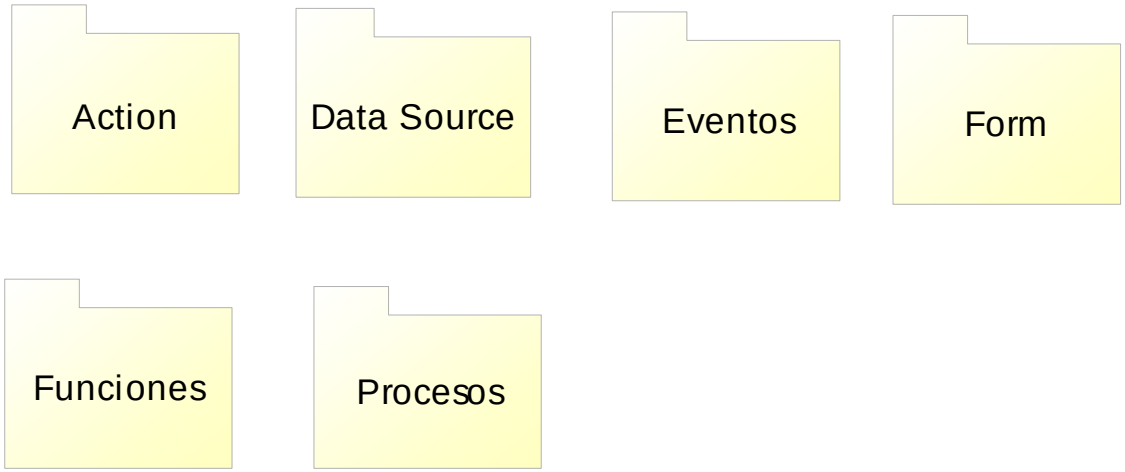



**Fig. 4.30** Diagrama de paquetes de clases

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

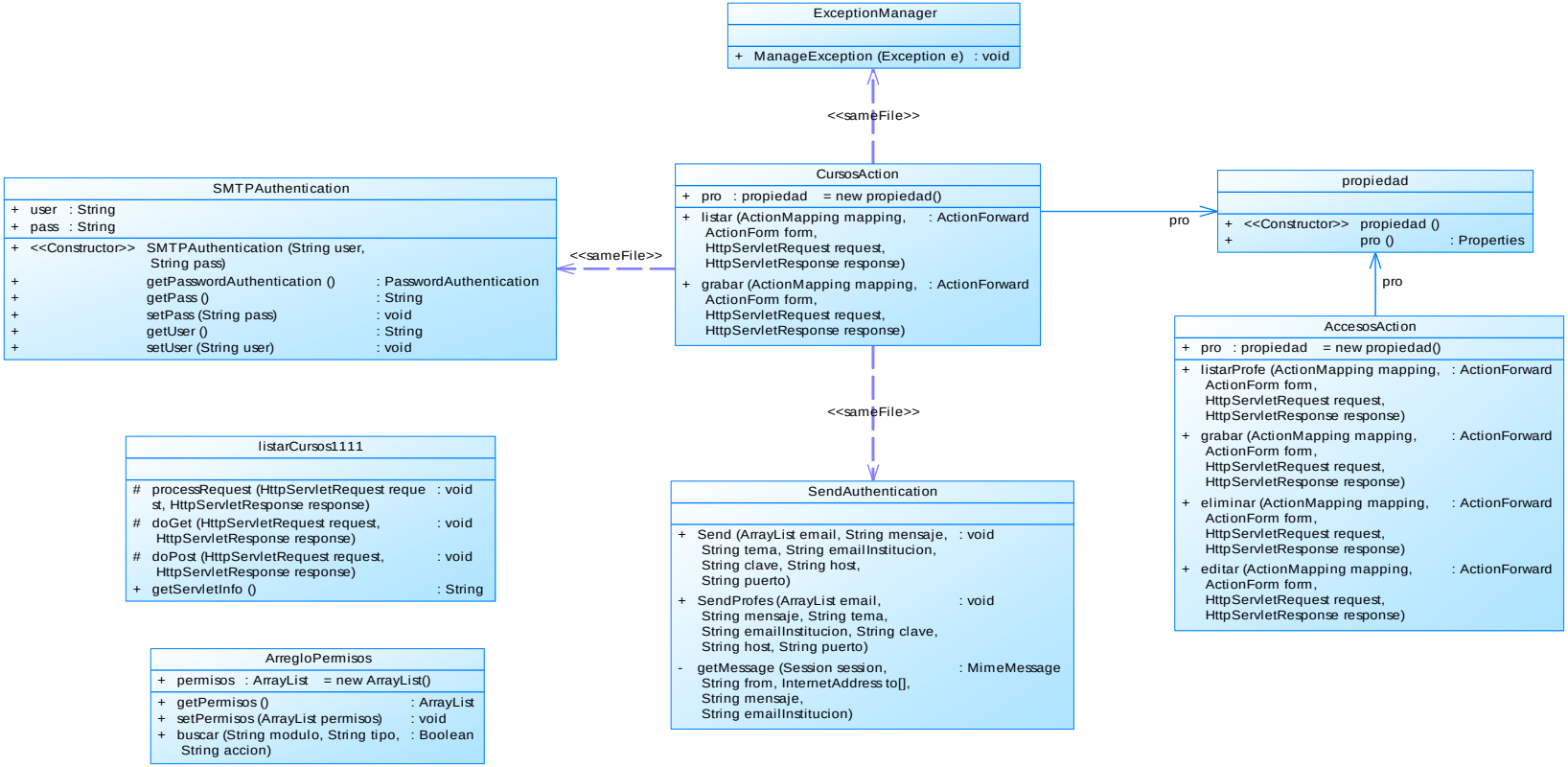



**Fig. 4.31** Diagrama de clases del paquete Action

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

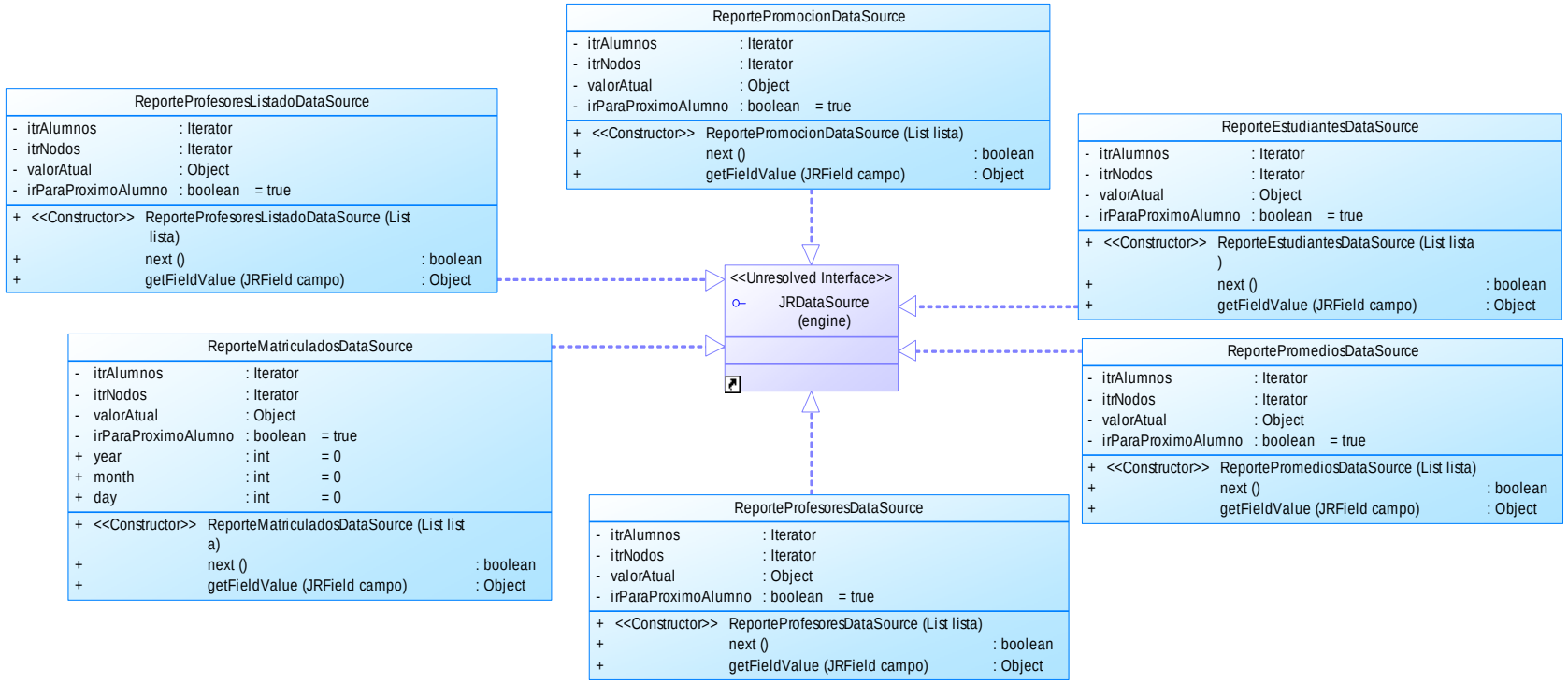



**Fig. 4.32** Diagrama de clases del paquete Data Source

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Fig. 4.33** Diagrama de clases del paquete Eventos

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

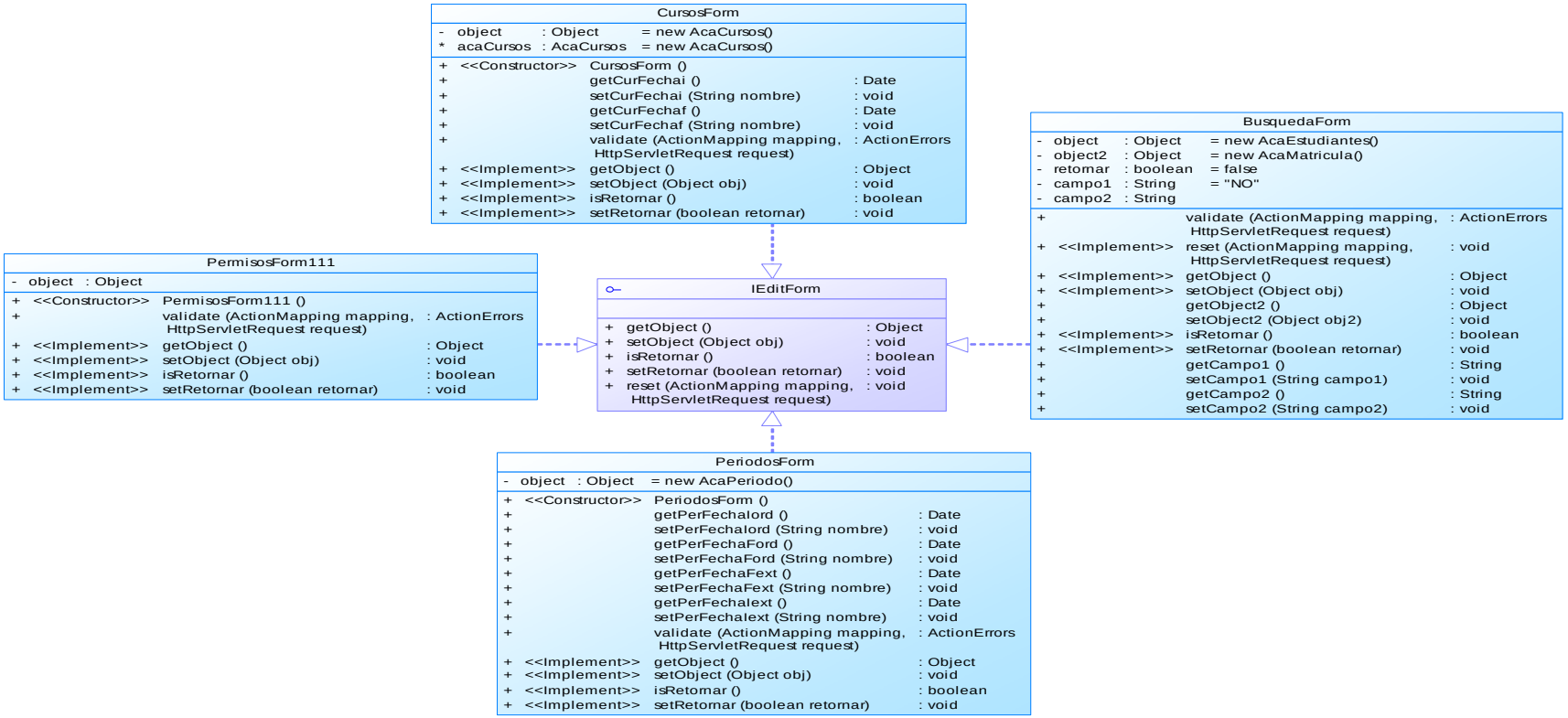



**Fig. 4.34** Diagrama de clases del paquete Form

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

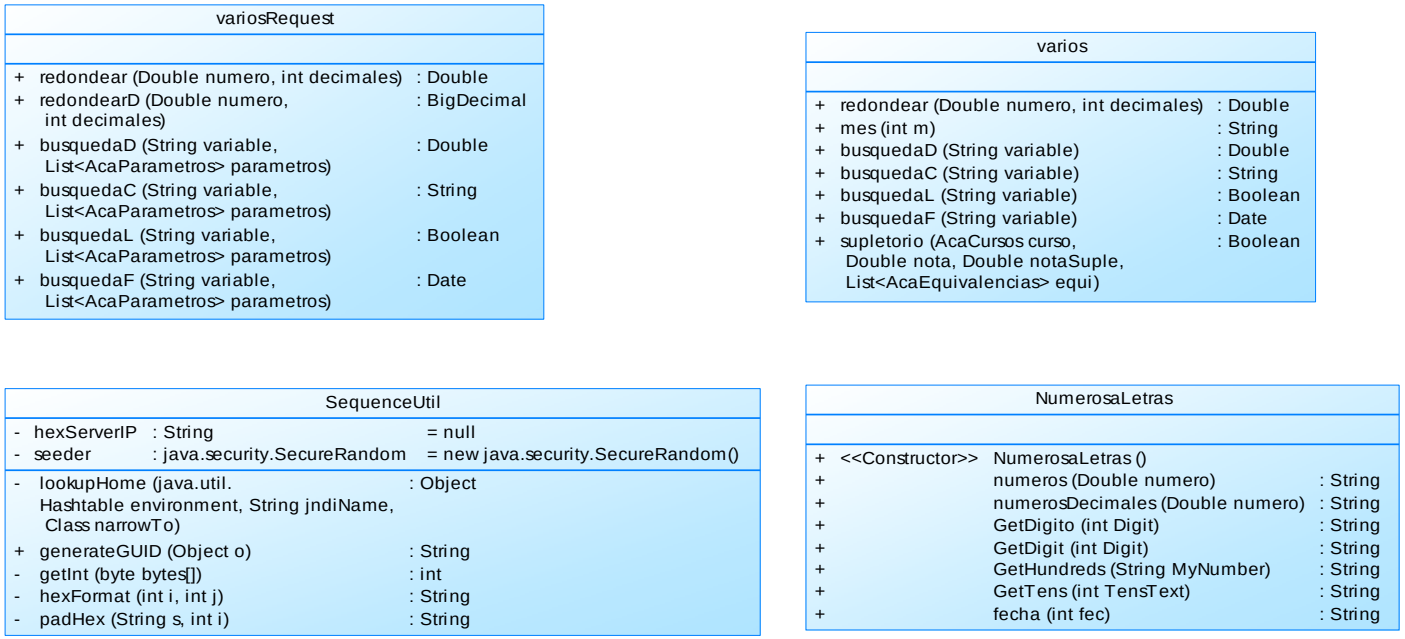



**Fig. 4.35** Diagrama de clases del paquete Funciones

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Fig. 4.36** Diagrama de clases del paquete Procesos

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

### 4.1.6 DISEÑO DE INTERFAZ

Las interfaces de usuario de cualquier sistema son muy importantes ya que es la parte con la que interactúa el usuario por lo que es clave que su diseño cumpla con las expectativas del mismo.

Hay que tener en cuenta que no todos los usuarios tienen los mismos conocimientos de manejo de aplicaciones por lo que se debe diseñar una interfaz que sea agradable con un usuario experto como de un principiante, esto se logra diseñando una interfaz totalmente estándar ya que si es cierto que no todas las pantallas pueden ser totalmente iguales, ya que cada una cumple con diferentes funciones, si se pueden igualar su estructura general.

Como parte de la estandarización de las interfaces aparecen unos conceptos muy importantes dentro del sistema.

***Richfaces****:* es una librería de código abierto basada en Java que permite crear aplicaciones web con AJAX.

Implementa sobre el JSF unos filtros para permitir peticiones AJAX en la página.

*Richfaces* está desarrollado y mantenido por *JBoss*; Este componente permite organizar menús jerárquicos similares al que casi todas las aplicaciones de escritorio tienen a más de barras de herramientas.

Para el diseño de las interfaces de usuario se ha utilizado Las reglas de oro[6] que son:

- **Dara el control al usuario:** se debe definir los módulos para que el usuario no tenga que realizar acciones no deseadas.

- **Reducir la carga de memoria de usuario:** esto le permite al usuario realice tareas en el contexto adecuado.

- **Construir una fase consecuente:** se debe presentar la información de manera consecuente, claro y precisa.

A continuación mostramos las estructuras generales para las pantallas que conforman el sistema:

---

- [6] Referencia bibliográfica: Pressman S, Roger: Ingeniería del Software un enfoque práctico. Quinta edición.


**Fig. 4.37** Pantalla de login

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

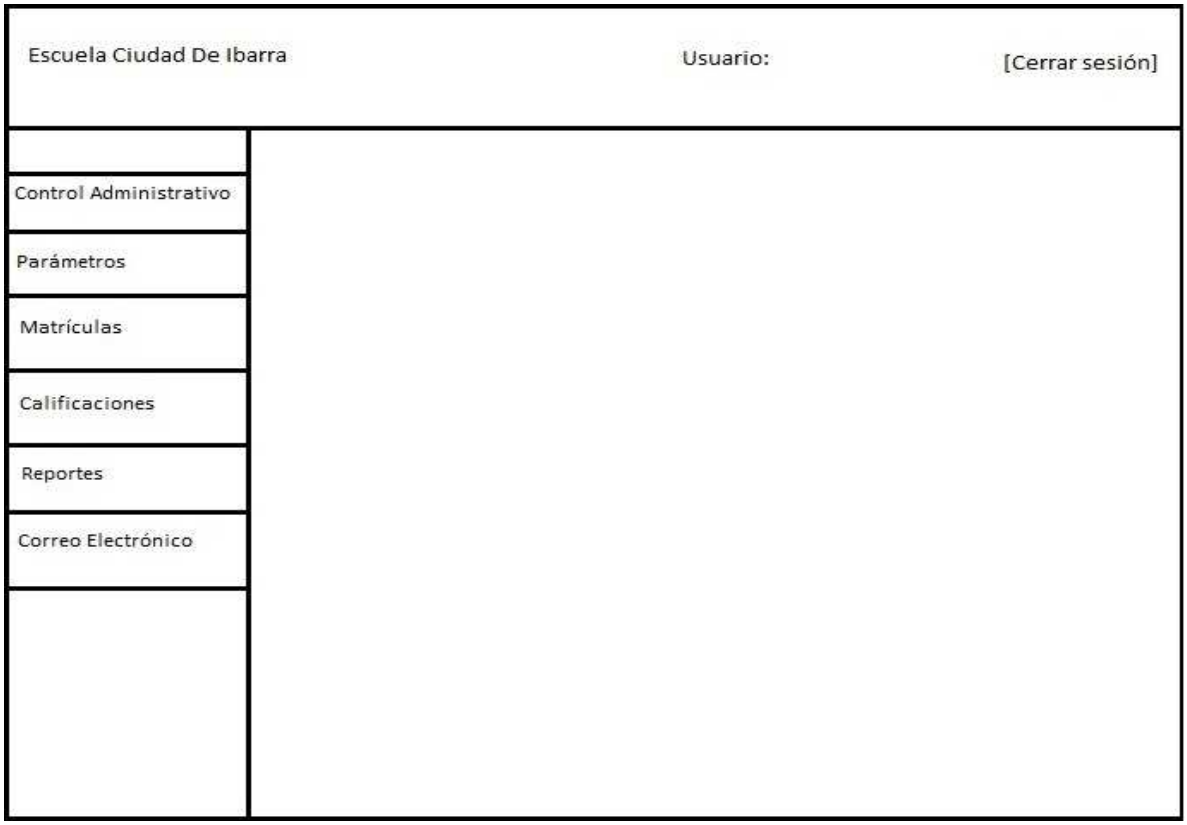



**Fig. 4.38** Pantalla de menú

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

# CAPÍTULO V

## 5. ETAPA DE CODIFICACIÓN

En esta se realiza el desarrollo de software es decir aquí se implementa el código fuente, haciendo uso de los datos obtenidos en fases anteriores.

Se utilizó programación orientada a objetos potencia el mantenimiento, la extensión la reutilización de código.

Este tipo de programación permite manejar:

Objetos: un objeto representa una entidad en la vida real. En la Fig. 5.1 se observar la implementación del objeto ”BeanAuditoria”.

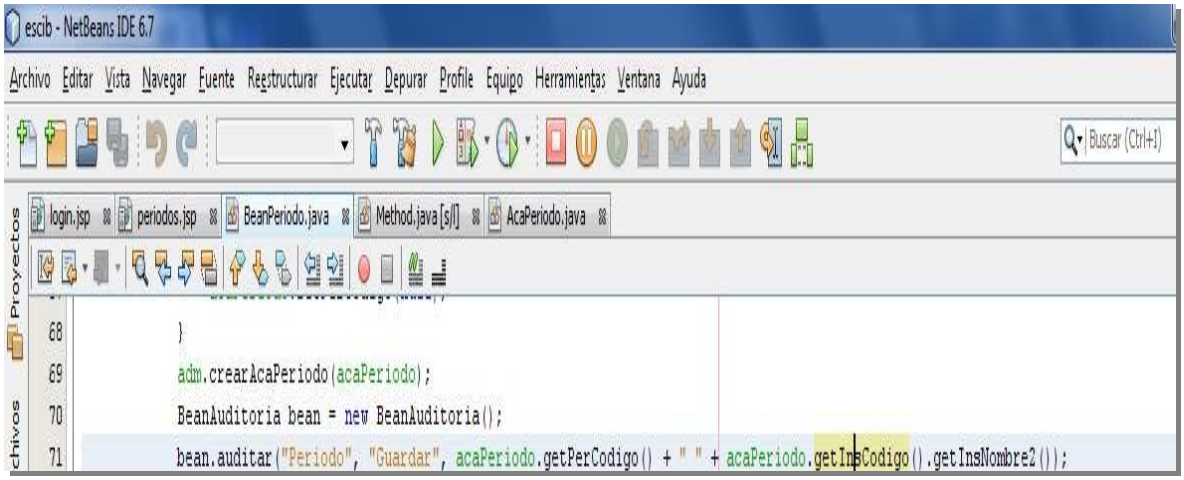



**Fig. 4.39** Ejemplo de objeto

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

Clases: define las propiedades y comportamiento de un objeto. En la figura 5.2 se observar la implementación de  la clase ”BeanPeriodo”.

```
escib - NetBeans IDE 6.7
Archivo Editar Vista Navegar Fuente Reestructurar Ejecutar Depurar Profile Equipo Herramientas Ventana Ayuda
login.jsp | periodos.jsp | BeanPeriodo.java | Method.java [s/l] | AcaPeriodo.java

20  */
21  public class BeanPeriodo {
22
23      private AcaPeriodo acaPeriodo;
24      private AcaPeriodo acaPeriodoDesde;
25      private AcaPeriodo acaPeriodoHasta;
26      private AdministradorSistema adm;
27      private String text;
28      private String texto2;
29      private String texto3;
30      private Date fecha;
31      private String modulo;
32
33      /** Creates a new instance of BeanPeriodo */
34      public BeanPeriodo() {
35          adm = new AdministradorSistema();
36          acaPeriodo = new AcaPeriodo();
37          mensajes = "";
38          inicializar();
39      }
40      public AcaPeriodo periodo = (AcaPeriodo) FacesContext.getCurrentInstance().getExternalContext().getSessionMap().get("periodoActual");
41      //FUNCION PARA LISTAR TODAS LAS ASIGNATURAS
42
43      public List<AcaPeriodo> getListaAsignaturas() {
44          return adm.listarAcaPeriodo();
45      }
46
47      public List<AcaPeriodo> getAcaPeriodos() {
48          return adm.listarAcaPeriodo();
49      }
50
51      public ArrayList<SelectItem> getPeriodos() {
```

**Fig. 4.40** Ejemplo de clase

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

Herencia: es la facilidad de que una clase herede los atributos y operaciones de otra clase.

Las librerías utilizadas en el sistema son las siguientes:

- Standar de java, propias de NetBeans

  Jstl1.1standar.jar

- Librerías de conexión para MySql

  *mysql-connector-java-5.0.6-bin.jar.*

- Librerías para JPA.

  *ejb3-persistence.jsr.*

  *toplink Essenciales-TopLink essenciales.jar.*

  *odbc14.jar.*

- Librerías para RichFaces.

  *RichFaces-api-3.2.1.GA.1.jar.*

- Librerías para reportes.

  *commons-beanutils-1.7.*

  *jasperreports-2.0.1.*

  *jasperreports-extensions-1.3.1.*

- Librerías para Ajax.

  *log4j-1.2.14.*

Para realizar los comentarios se utilizó el siguiente formato:

- // Descripciones especificas.
- /* Fecha

  Nombre de la persona que modificó

  Descripción*/

Todo el código fuente se encuentra en el CD adjunto a este proyecto.

## 6. ETAPA DE IMPLEMENTACIÓN

En esta etapa se pone en funcionamiento el sistema, se aplican los métodos y medidas necesarias para llevar a cabo el objetivo, el mismo que debe pasar por un período de prueba para ir depurando la aplicación hasta que su funcionalidad se encuentre al 100%, garantizando su uso óptimo al momento de realizar los procesos respectivos.

Se implementó un piloto del sistema para realizar las depuraciones necesarias, garantizando su funcionalidad en el período de matriculación e ingreso a clases de los estudiantes, el cual sería también para realizar las pruebas de software.

Los pasos a seguir para implementar la aplicación ESCIB en la Escuela Fiscal Mixta Ciudad de Ibarra son los siguientes:

- Instalar *Apache Tomcat 6.0* en un servidor Windows 2003 server o superior de preferencia que tenga Windows XP.

- Instalar JDK 6.0

- Publicar la aplicación escib.war dentro de la carpeta webapp del servidor de aplicaciones.

- Instalar MSql 6.0, MySql Administrator con la configuración típica e ingresar usuario y contraseña con *root* respectivamente.

- Restaurar la base de datos aca_demico.

- Verificar que se encuentre instalado un navegador web de preferencia *Google Chrome* o *Mozilla Firefox*.

- Verificar que *Apache Tomcat 6.0* se encuentre iniciado su servicio en la ruta Panel de control\Todos los elementos de Panel de control\Herramientas administrativas\servicios.

- Ingresar a la dirección htpp://192.168.4.1/escib, para verificar que se encuentre correcta la publicación del sistema.

# 7. ETAPA DE PRUEBAS

Son aquellos procesos que permiten verificar y revelar la calidad del software. Para realizar pruebas a un software lo ideal es utilizar técnicas que nos faciliten descubrir los errores.

Dichas pruebas deben permitir comprobar el grado de cumplimiento respecto a las especificaciones iniciales y permitiendo detectar errores o fallas y asegurarnos de que el software cumple con los conceptos de estabilidad, escalabilidad, eficiencia y seguridad que se relacionan a la calidad una aplicación.

Para el sistema académico ESCIB se van a aplicar pruebas de caja negra.

## 7.1 PRUEBA DE CAJA NEGRA

Las pruebas de caja negra permiten obtener condiciones de entrada que ejecutan los requisitos funcionales del programa, estas se llevan a cabo a partir de la interfaz del programa, obviando el comportamiento interno y la estructura del programa.

Las principales características que ofrecen estas pruebas son las siguientes:

- Las funciones del software se encuentran operativas.
- La entrada se encuentra de forma adecuada y la salida de información es correcta.
- La integridad de la información externa se mantiene correcta.

**PRUEBA INGRESO DE USUARIOS**

| Condición de Entrada | Tipo | Clase Equivalencia Válida | Clase Equivalencia No Válida |
|---|---|---|---|
| Usuario | Valor específico | 1) Letras, caracteres especiales y número | 2) Campo vacío. |
| Contraseña | Valor específico | 3) Letras, caracteres especiales y números hasta 20 caracteres | 4) Campo vacío. |

**Tabla 5.1.** Condiciones de Entrada – Caja Negra

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

| Equivalencia | Caso Prueba | Respuesta Esperada | Resultado Obtenido | Conclusiones |
|---|---|---|---|---|
| **1** | 1) Usuario: “amora”<br>2) Contraseña: “xxx” | Permite escoger el año lectivo. | 1) Continúa el proceso con éxito. | El algoritmo verifica que el usuario y la contraseña estén registrados en la base de datos, caso contrario envía un mensaje de error. |
| **2** | 1)Usuario: “prueba”<br>2)Contraseña: “xxxx” | No permite continuar | 2) Se desplegó mensaje de error: “Usuario incorrecto” | Validación realizada es correcta ya que solo permite usuario y contraseñas registradas en la base de datos. |

| Equivalencia | Caso Prueba | Respuesta Esperada | Resultado Obtenido | Conclusiones |
|---|---|---|---|---|
| **3** | 1) Usuario: “mehernan”<br>2) Contraseña: “xxxx” | Continua el proceso, muestra el menú principal | 1) El sistema verifica el perfil del usuario y permite ver todo los submenús asignados al perfil. | Si el usuario tiene un perfil de administrador le permite ver los menús con sus submenús completos. |
| **4** | 1) Usuario: “chernan”<br>2) Contraseña: “xxxx” | Permite ingresar al menú principal y mostrar los submenús | 1) El sistema verifica el perfil del usuario y restringe los menús y submenús de acuerdo al perfil asignado, ya que tiene perfil de docente. | Si el usuario tiene un perfil distinto al de administrador el sistema restringe el acceso a los diferentes menús y submenús. |

**Tabla 5.2.** Condición de Prueba - Caja Negra

Fuente: Autores de Tesis

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**PRUEBA MÓDULO DE MATRICULACIÓN**

| Condición Entrada | Tipo | Clase Equivalencia Válida | Clase Equivalencia No Válida | Conclusión |
|---|---|---|---|---|
| 1)Identificación | Valor específico | 5) Cédula: Igual a 10 dígitos. | Menor a 13 dígitos<br>Letras<br>Caracteres especiales | Se validó correctamente. |
| 2)Apellidos | Valor específico | Cadena de caracteres | Espacio en blanco | Se validó correctamente. |
| 3)Nombres | Valor específico | Cadena de caracteres | Espacio en blanco | Se validó correctamente. |
| 4)Dirección | | Cadena de caracteres | | Se validó correctamente. |
| 5)Teléfonos | | Cadena de caracteres | Solo números | Se validó correctamente. |
| 6)Movil | | Cadena de caracteres | Solo números | Se validó correctamente. |

| Condición Entrada | Tipo | Clase Equivalencia Válida | Clase Equivalencia No Válida | Conclusión |
|---|---|---|---|---|
| 7)Nivel a matricular | | Seleccionar | | Se validó correctamente. |
| 8)Fecha Nacimiento | Valor específico | Fecha en formato: año-mes-día | 17) Fechas mayores a la actual. | El proceso no respondió según lo esperado. |
| 9)Lugar de nacimiento | | Fecha en formato: año-mes-día | | Se validó correctamente. |
| 10)Foto | Valor específico | Imagen | 20) Imágenes en formato .jpg | No permite otros formatos para subir fotografías. |
| 11)Género | Miembro de un conjunto | FEMENINO MASCULINO | | Se validó correctamente. |
| 12)Instituto anterior | Miembro de un conjunto | Cadena de caracteres | | Se validó correctamente. |
| 13)Aprovechamiento | Valor específico | Números | | No paso el proceso de validación. |

| Condición Entrada | Tipo | Clase Equivalencia Válida | Clase Equivalencia No Válida | Conclusión |
|---|---|---|---|---|
| 14)Email | Valor específico | Cadena con @ seguido de un dominio (.com) | Espacio en blanco<br>Cadena sin @<br>Dirección sin dominio | Se validó correctamente. |
| 15)Fecha matricula | Valor específico | Fecha en formato: año-mes-día | | El proceso no respondió según lo esperado. |
| 16)Nacionalidad | Valor específico | Cadena de caracteres | Espacio en blanco | Se validó correctamente. |
| 17)Datos del padre | Valor específico | Cadena de caracteres | | El proceso no respondió según lo esperado. |
| 18)Datos de la madre | Valor específico | Fecha en formato: año-mes-día | Letras o caracteres especiales | El proceso no respondió según lo esperado. |

| Condición Entrada | Tipo | Clase Equivalencia Válida | Clase Equivalencia No Válida | Conclusión |
|---|---|---|---|---|
| 19)Representante | Miembro de un Valor especifico | Cadena de caracteres | | El proceso no respondió según lo esperado. |

**Tabla 5.3.** Condición de Entrada - Caja Negra

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

# 8. ETAPA DE MANTENIMIENTO

En la mayoría de las ocasiones el software sufre cambios después de que se entrega al cliente. Estos cambios pueden ser para corregir errores o para mejorar el sistema.

Se puede hablar de tres tipos de mantenimiento que se realizara sobre el sistema:

## 8.1 MANTENIMIENTO CORRECTIVO

Tiene como objetivo eliminar problemas ocasionados durante la fase operativa del sistema, por lo general estos errores son reportados por los usuarios.

Por el momento se aplicará solamente esta clase de mantenimiento, puesti que el sistema se encuentra en fase piloto hasta que empiece el periodo de matriculación e ingreso a clases.

La Fig. 5.1 muestra el proceso a seguir cuando se tenga un error en el sistema, el formulario para el envío de errores véase en el Anexo 6.

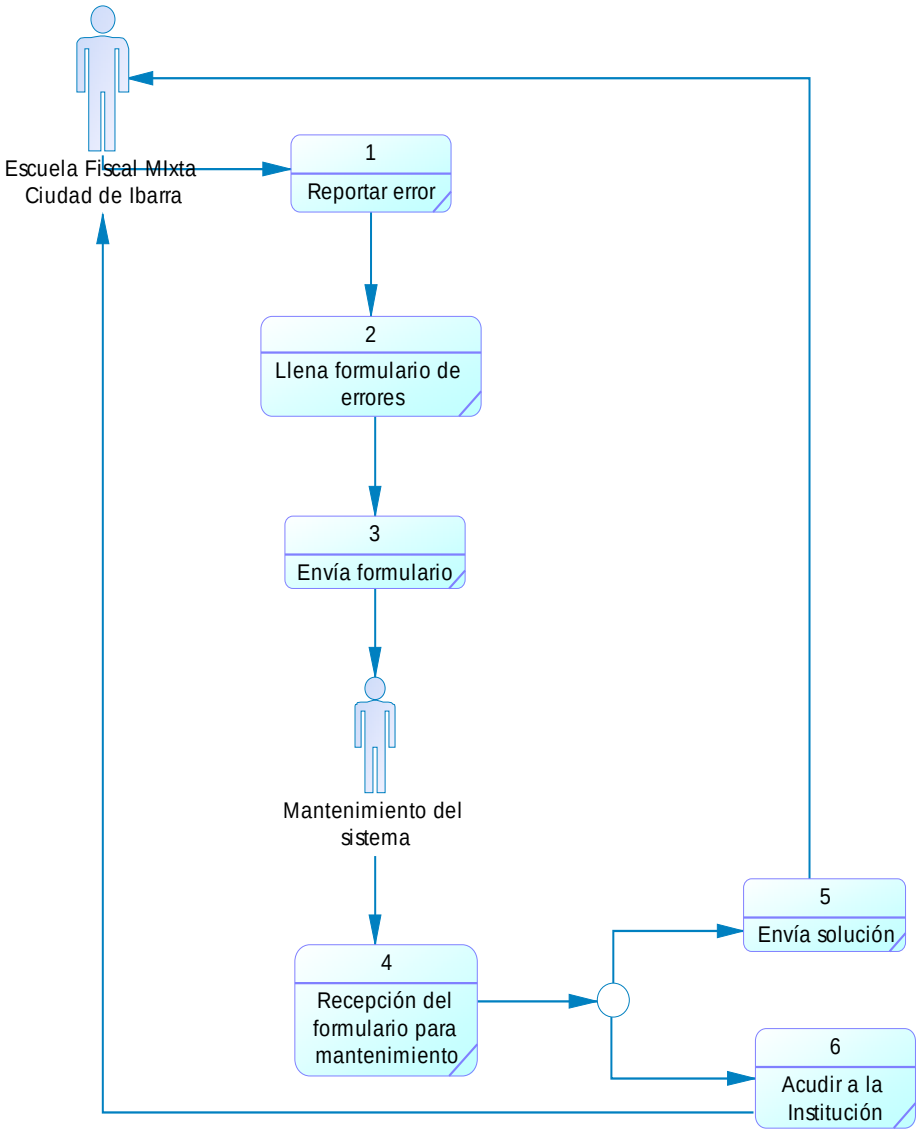



**Fig. 5.1.** Reportar errores

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

## 8.2 MANTENIMIENTO PERFECTIVO

Tiene como objetivo mejorar la funcionalidad del sistema por ejemplo en la eficiencia en ejecución, facilitar su uso, etc.

## 8.3 MANTENIMIENTO EVOLUTIVO

Este tipo de mantenimiento por lo general se realiza después de puesto en producción el sistema ya que nos permite ampliar, eliminar o sustituir la funcionalidad del mismo según cambien o aumenten las necesidades del usuario.

# 9. CONCLUSIONES

- Se implemento una aplicación de tipo web para la institución que permite a los docentes realizar su trabajo con agilidad, facilidad mejorando así la calidad de su trabajo y acortando los tiempos estimados para llevar a cabo los procesos establecidos.

- A partir del análisis de los requerimientos solicitados por el usuario se generaron reportes que facilitan el trabajo realizado por los docentes en sus diferentes roles, los mismos que optimizan la entrega de calificaciones, listas de alumnos, estudiantes matriculados, entre otros.

- La información se manejada con seguridad por el uso de perfiles dentro del sistema educativo la misma que es almacenada en un repositorio de base de datos con lo cual se ahorra papel contribuyendo con el medio ambiente y evitando la pérdida de está.

- Con la aplicabilidad de los módulos la información se consolidó en un sistema académico vía web para la Escuela Fiscal Mixta Ciudad de Ibarra, evitando así tener la información dispersa en diferentes fuentes,

facilitando a los docentes el acceso a los datos en cualquier momento del día.

- ESCIB fue desarrollado pensando en los docentes de la Institución, mostrando así interfaces amigables, sencillas para su uso, de esta manera el usuario final no se sentirá confundido y realizará su trabajo con facilidad como lo hace con cualquier aplicación ofimática mejorando su calidad de trabajo.

- Las tecnologías utilizadas para el desarrollo del sistema educativo como AJAX, JPA y JSF permiten mejorar la calidad del mismo agilitando los tiempos de respuesta para cada proceso ejecutado por los usuarios de la aplicación.

- Se utilizó software libre ya que la Escuela Fiscal Mixta Ciudad de Ibarra no posee los recursos necesarios para invertir en herramientas costosas, motivo por el cual se escogió Netbeans 6.7 y Mysql.

# 10. RECOMENDACIONES

- Se recomienda a los docentes realizar una capacitación antes del periodo establecido para la matriculación de los estudiantes, con la cual se despejará dudas y problemas al momento de realizar este proceso.

- La información se encuentra almacenada en un solo repositorio por lo cual se recomienda realizar respaldos cada cierto tiempo y guardarlos en otro dispositivo de ser posible con esto se preverá cualquier pérdida de datos.

- Se recomienda documentar cada solución que se dé a alguna falla presentada en el sistema para mantener un historial y acudir a él en caso de volver a suceder otra vez.

- Se recomienda llenar el formulario respectivo para reportar los errores que se den en el sistema educativo de manera clara y concisa para dar solución a los mismos de manera rápida y efectiva.

- Se recomienda una buena estructura de la red para evitar la latencia en el uso del sistema, la máquina que va ha ser servidor de aplicación debe cumplir con condiciones óptimas la mismas que garanticen el funcionamiento del sistema educativo.

- Finalmente es importante entender que ninguna red ni programa proporciona seguridades al 100%. Para evitar problemas de la manipulación de información se recomienda tratar de incorporar seguridades que garanticen integridad, confidencialidad y disponibilidad.

# 11. BIBLIOGRAFÍA

## Libros


- Mendelzon, Alberto; Ale, Juan: Introducción a las Bases de Datos relacionales. Buenos Aires: Prentice Hall, 2000.

- Ullman, Jeffrey D.; Widom, Jennifer: Introducción a los sistemas de base de datos. México: Prentice Hall, 1999.

- Liza Ávila, César: Modelado con UML, Principios Aplicaciones. Primera Edición, Agosto 2001.

- Pressman S, Roger: Ingeniería del Software un enfoque práctico. Quinta edición.

**Páginas web**


- Normalización de base de datos
http://es.wikipedia.org/wiki/Normalizaci%C3%B3n_de_bases_de_datos

- Desarrollo en cascada
http://es.wikipedia.org/wiki/Desarrollo_en_cascada#An.C3.A1lisis_de_requisitos

- Modelo de datos
http://bd.eui.upm.es/BD/docbd/tema/tema2.pdf

- Diseño de base de datos relacionales
http://www.mailxmail.com/curso-diseno-base-datos-relacionales/diseno-logico-bases-datos

- Crear sitios y aplicaciones web en JSP con Java
http://www.cristalab.com/tutoriales/crear-sitios-y-aplicaciones-web-en-jsp-con-java-c74730l/

- JavaServer Pages Technology
http://www.oracle.com/technetwork/java/javaee/jsp/index.html

- Desarrollo de Aplicaciones Web con JSP y XML
  http://www.programacion.com/articulo/desarrollo_de_aplicaciones_web_con_jsp_y_xml_120

- JavaServer Pages
  http://es.wikipedia.org/wiki/JavaServer_Pages
- Programación por capas
  http://es.wikipedia.org/wiki/Programaci%C3%B3n_por_capas

- Directivas JSP
  http://artemisa.unicauca.edu.co/~dparedes/java/jsp/jspbasics.fm0.html

- AJAX
  http://es.wikipedia.org/wiki/AJAX

- JavaServer Faces
  http://es.wikipedia.org/wiki/JavaServer_Faces

- JavaServer Faces Technology
  http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html

- Introducción a JSF Java Server Faces
  http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=IntroduccionJSFJava

# ANEXO 1

ENCUESTA AL PERSONAL DOCENTE Y ADMINISTRATIVO DE LA INSTITUCIÓN

**ENCUESTA**

**“ESCUELA FISCAL MIXTA CIUDAD DE IBARRA”**

La presente encuesta tiene como objetivo recopilar información acerca de los procesos actuales con los que se maneja la institución para realizar un sistema académico que automatice dichos procesos.

1.- ¿Cuántas personas realizan las tareas administrativas?

De 1 a 10 ☐

Más de 10 ☐

Menos de 10 ☐

2.- ¿Cuál es el número de docentes?

De 1 a 10 ☐

Más de 10 ☐

Menos de 10 ☐

3.- ¿Cómo es el proceso para entregar el reporte de calificaciones trimestralmente?

Libretas (Manualmente) ☐

Hojas impresas (Automático) ☐

4.- ¿Cómo es el proceso para entregar cualquier otro tipo de reporte administrativo?

A través de un sistema académico (Automático) ☐

Por medio de formatos establecidos (Máquina de escribir) ☐

5.- ¿Qué proceso realiza el docente para pasar las notas al personal administrativo?

Hojas de cálculo ☐

Hojas llenadas a mano ☐

Vía web ☐

6.- ¿Cuál es el proceso que deben seguir al matricularse?

Por medio de un sistema académico ☐

Formularios llenados a mano ☐

7.- ¿Cuál es el tiempo que se demora en matricular por alumno?

15 a 30 min ☐

30 a 45 min ☐

Más de 1 hora ☐

# ANEXO 2

DICCIONARIO DE DATOS

**BASE DE DATOS aca_demico**

**Tabla aca_accesos**

Describe los permisos para cada usuario el sistema académico.

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|
| acc_codigo | Int | Primary key | Clave principal de la tabla aca_accesos |
| acc_modulo | varchar(3) | | Define los modulos que hay en el sistema ademico |
| acc_ingresar | tinyint | | Define con 1 y 0 si desea que se realice la operación. |
| acc_agregar | tinyint | | Define con 1 y 0 si desea que se realice la operación. |
| acc_modificar | tinyint | | Define con 1 y 0 si desea que se realice la operación. |
| acc_eliminar | tinyint | | Define con 1 y 0 si desea que se realice la operación. |
| acc_perfil | varchar(3) | | Define el perfil con el que se ingreso al sistema. |

DATOS TABLA aca_accesos
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_anio**

Describe el año para posteriormente relacionarse con otra tablas.

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|
| ani_codigo | int | Primary key | Clave primaria de la tabla aca_anio |
| ani_nombre | int | | Se almacenan los años. |

DATOS TABLA aca_anio
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_asignatura**

Describe las asignaturas que posee la Escuela Ciudad de Ibarra

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|
| asi_codigo | int | Primary key | Clave primaria de la tabla asi_codigo. |
| asi_nombre | varchar(70) | | Describe el nombre de la asignatura. |
| asi_abreviatura | varchar(10) | | Describe la abreviación de la asignatura. |
| asi_tipo | varchar(1) | | Describe el tipo de asignatura. |

DATOS TABLA aca_asignatura
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_auditoria**

Describe los campos para entregar los resultados de auditoría en el módulo Control administrativo.

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|

| | | | |
|---|---|---|---|
| aud_codigo | Int | Primary key | Clave primaria de la tabla aca_auditoria |
| aud_usuario | int | | Describe el usuario que realizó el proceso. |
| aud_ip | varchar(20) | | Describe la IP de la máquina desde donde se ejecutó el proceso. |
| aud_maquina | varchar(20) | | Describe el nombre de la máquina donde se ejecutó el proceso. |
| aud_tabla | varchar(50) | | Describe que proceso realizó. |
| aud_tipo | varchar(20) | | Describe tipo de proceso que ejecutó. |
| aud_materiaprofe | varchar(100) | | Describe el usuario que realizó el proceso. |
| aud_fecha | datetime | | Describe la fecha en la que fue realizado el proceso. |

DATOS TABLA aca_auditoria
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_cursos**

Describe los cursos y las notas de la Escuela Ciudad de Ibarra.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| cur_codigo | Int | Primary key | Calve principal de la tabla aca_cursos |
| per_codigo | Int | Foreign key | Clave primaria de la tabla aca_periodo |

| | | | |
|---|---|---|---|
| ncur_codigo | Int | Foreign key | Clave primaria de la tabla aca_ncurso |
| par_codigo | Int | Foreign key | Clave primaria de la tabla aca_paralelo |
| esp_codigo | Int | Foreign key | Clave primaria de la tabla aca_especialidad |
| cur_cupos | Int | | Define el número de cupos por curso. |
| cur_fechai | Date | | Define la fecha de inicio del curso. |
| cur_fechaf | Date | | Define la fecha final del curso. |
| cur_notaprueba | Double | | Define la nota por prueba. |
| cur_notaminima | Double | | Define la nota mínima en el curso. |
| cur_secuencianivel | Int | | Define la secuencia del curso. |
| cur_insnombre | varchar(100) | | Define el nombre del inspector del curso. |
| cur_dirnombre | varchar(100) | | Define el nombre del director del curso. |
| cur_inscedu | varchar(13) | | Define la cedula |
| cur_dircedu | varchar(13) | | Define la cedula |
| cur_seccion | varchar(30) | | Define la sección a la que pertenece el curso. |

DATOS TABLA aca_cursos
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_disciplina**

Describe las respectivas notas de disciplina por materia.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| dis_codigo | Int | Primary key | Clave primaria de la tabla aca_disciplina. |
| mat_codigo | Int | Foreign key | Clave principal de la tabla aca_matricula. |
| sis_codigo | Int | Foreign key | Clave principal de la tabla aca_sistemanotas. |
| dis_fecha | Date | | Define la fecha. |
| dis_n1 | Int | | Defina el valor del la nota de disciplina. |
| dis_n2 | Int | | Defina el valor del la nota de disciplina. |
| dis_n3 | Int | | Defina el valor del la nota de disciplina. |
| dis_n4 | Int | | Defina el valor del la nota de disciplina. |
| dis_n5 | Int | | Defina el valor del la nota de disciplina. |
| dis_n6 | Int | | Defina el valor del la nota de disciplina. |
| dis_n7 | Int | | Defina el valor del la nota de disciplina. |
| dis_n8 | Int | | Defina el valor del la nota de disciplina. |
| dis_n9 | Int | | Defina el valor del la nota de disciplina. |

| | | | |
|---|---|---|---|
| dis_n10 | Int | | Defina el valor del la nota de disciplina. |
| dis_observacion | varchar(500) | | Define las observaciones si son necesarias. |

DATOS TABLA aca_disciplina
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_equivalencias**

Describe las equivalencias para las respectivas notas.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| eq_codigo | Int | Primary key | Clave primaria de la tabla aca_equivalencias |
| eq_grupo | varchar(2) | | Define al grupo que pertenece la equivalencia de la nota. |
| eq_nombre | varchar(50) | | Define el nombre de la equivalencia de la nota. |
| eq_abreviatura | varchar(10) | | Define la abreviatura de la equivalencia de la nota. |
| eq_vminimo | Double | | Define el valor mínimo de la equivalencia de la nota. |
| eq_vmaximo | Double | | Define el valor máximo de la equivalencia de la nota. |
| per_codigo | int | Foreign key | Clave principal de la tabla aca_periodo. |

DATOS TABLA aca_equivalencias
FUENTE: Tesista(s)

AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_especialidad**

Describe las especialidades de las materias.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| esp_codigo | int | Primary key | Clave primaria de la tabla aca_especialidad |
| esp_nombre | varchar(100) | | Define el nombre de la especialidad. |

DATOS TABLA aca_especialidad
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_estudiantes**

Describe los datos de los estudiantes.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| est_codigo | Int | Primary key | Clave principal de la tabla aca_estudiantes. |
| est_nombre | Varchar(50) | | Define el nombre del estudiante. |
| est_apellido | Varchar(50) | | Define el apellido del estudiante. |
| est_direccion | Varchar(100) | | Define la dirección del estudiante. |
| est_telefono1 | Varchar(50) | | Define teléfono del estudiante. |

| | | | |
|---|---|---|---|
| est_telefono2 | Varchar(50) | | Define teléfono del estudiante. |
| est_movil | Varchar(50) | | Define teléfono móvil del estudiante. |
| est_mail1 | Varchar(100) | | Define mail del estudiante. |
| est_mail2 | Varchar(100) | | Define mail del estudiante. |
| est_tipoId | Varchar(1) | | Define el tipo de identificación del estudiante. |
| est_identificacion | Varchar(13) | | Define la identificación del estudiante. |
| est_fechaNac | Date | | Define fecha de nacimiento del estudiante. |
| est_genero | Varchar(1) | | Define género del estudiante. |
| est_nacionalidad | varchar(50) | | Defina nacionalidad del estudiante. |

DATOS TABLA aca_estudiantes
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_representantes**

Describe los datos de los representantes.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| re_codigo | Int | Primary key | Clave principal de la tabla aca_estudiantes. |

| | | | |
|---|---|---|---|
| est_codigo | Int | Foreing key | Clave foránea estudiantes. |
| re_nombre | Varchar(50) | | Define el nombre del representante. |
| re_apellido | Varchar(50) | | Define el apellido del representante. |
| re_direccion | Varchar(100) | | Define la dirección del representante. |
| re_telefono1 | Varchar(50) | | Define teléfono del representante. |
| re_telefono2 | Varchar(50) | | Define teléfono del representante. |
| re_movil | Varchar(50) | | Define teléfono móvil representante. |
| re_mail1 | Varchar(100) | | Define mail del representante. |
| re_mail2 | Varchar(100) | | Define mail del representante. |
| est_identificacion | Varchar(13) | | Define la identificación del representante. |

DATOS TABLA aca_representante

FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_hermanos**

Describe si el estudiante tiene hermanos en la institución.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|

| | | | |
|---|---|---|---|
| her_codigo | Int | Primary key | Clave primaria de la tabla aca_hermanos. |
| est_codigo | Int | Foreign key | Clave primaria de la tabla aca_estudiantes. |
| her_estado | Tinyint | | Define el estado. |
| her_principal | Tinyint | | Define si es principal. |

DATOS TABLA aca_hermanos
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_institucion**

Describe los datos de la Escuela Ciudad de Ibarra.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| ins_codigo | Int | Pimary key | Clave principal de la tabla aca_institucion. |
| ins_siglas | Varchar(50) | | Define las siglas para la institución. |
| ins_nombre | Varchar(200) | | Define el nombre de la institución. |
| ins_nombre2 | Varchar(100) | | Define el nombre de la institución. |
| ins_direccion | Varchar(100) | | Define la dirección de la institución. |
| ins_lugar | Varchar(100) | | Define el lugar de la institución. |
| ins_telefono1 | Varchar(13) | | Define el teléfono de la institución. |

| | | | |
|---|---|---|---|
| ins_telefono2 | Varchar(13) | | Define el teléfono de la institución. |
| ins_fax | Varchar(13) | | Define el fax de la institución. |
| ins_email | Varchar(100) | | Define mail de la institución. |
| ins_rector | Varchar(100) | | Define el nombre del rector de la institución. |
| ins_secretaria | Varchar(100) | | Define el nombre de la secretaria de la institución. |
| ins_inspector | Varchar(100) | | Define el nombre del inspector de la institución. |
| ins_sello | Longblob | | Define el sello de la institución como imagen. |
| ins_fondo | Longblob | | Define el fondo de la institución como imagen. |
| ins_fondoCarne | longblob | | Define el fondo de carnet de la institución como imagen. |
| ins_clave | Varchar(100) | | Define la clave de la institución. |
| ins_puerto | Varchar(100) | | Define el puerto |

DATOS TABLA aca_institucion
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_jornada**

Describe las jornadas de estudio.

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|
| jor_codigo | Int | Primary key | Clave primaria de la tabla aca_jornada |
| jor_nombre | Varchar(50) | | Define el nombre de la jornada. |

DATOS TABLA aca_jornada
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_materiaprofesor**

Describe las materias por profesor.

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|
| map_codigo | Int | Primary key | Calve primaria de la tabla. aca_materiaprofesor |
| cur_codigo | Int | Foreign key | Clave principal de la tabla aca_cursos. |
| asi_codigo | Int | Foreign key | Clave principal de la tabla aca_asignatura. |
| pro_codigo | Int | Foreign key | Clave principal de la tabla aca_profesor. |
| map_cuantitativa | Tinyint | | Define si la materia va a ser cuantitativa. |
| map_opcional | Tinyint | | Define si la materia es opcional. |
| map_formula | varchar(100) | | Define la fórmula de la |

| | | | |
|---|---|---|---|
| | | | materia. |
| map_seimprime | Tinyint | | Define si se va imprimir la materia |
| map_control | Tinyint | | Define control. |
| map_orden | Int | | Define orden. |
| map_bloque | Tinyint | | Define bloque. |

DATOS TABLA aca_materiaprofesor
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_matrícula**

Describe los datos necesarios para el módulo de matrículas.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| mat_codigo | Int | Primary key | Clave primaria de la tabla aca_matricula |
| est_codigo | Int | Foreign key | Clave primaria de la tabla aca_estudiantes |
| cur_codigo | Int | Foreign key | Clave primaria de la tabla aca_cursos |
| mat_matricula | Int | | Define matrícula. |
| mat_fechaMat | Date | | Define la fecha de la matricula. |
| mat_fechaIns | Date | | Define la fecha de inscripción. |
| mat_fechaRet | Date | | Define la fecha |
| mat_fechaCam | Date | | Define la fecha |

| | | | |
|---|---|---|---|
| mat_folio | Int | | Define el folio de la matricula. |
| mat_estado | Int | | Define el estado de la matricula. Revisar con los datos de la base |
| mat_numeroMat | Int | | Define el número de matrícula. |
| mat_observacion1 | Varchar(100) | | Define las observaciones |
| mat_observacion2 | Varchar(100) | | Define las observaciones. |
| mat_aprobado | Tinyint | | Define si esta aprobado. |
| mat_estadoMat | Varchar(1) | | Define el estado de la matricula. |
| mat_foto | Blob | | Define la foto del estudiante. |
| mat_secretaria | Tinyint | | Define la secretaria que realizó el proceso de matriculación. |
| mat_guia | Varchar(7) | | Define la guía |
| mat_fecha | Date | | Define la fecha de matriculación. |
| mat_nueva | Tinyint | | Define si es matricula nueva |
| mat_valor | Double | | Define el valor de la matricula. |
| mat_matriculadopor | Varchar(200) | | Define la persona que realizo el proceso de matriculación. |

DATOS TABLA aca_matricula
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_nomcurso**

Describe los cursos que existen en la Escuela Ciudad de Ibarra

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| ncur_codigo | Int | Primary key | Clave primaria de la tabla aca_nomcurso |
| ncur_nombre | Varchar(100) | | Define el nombre del curso. |

DATOS TABLA aca_nomcursos
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_notanotas**

Describe las notas.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| | | | |
| Codigo | Int | Primary key | Clave primaria de la |

| | | | |
|---|---|---|---|
| | | | tabla aca_nomcurso. |
| | | | |
| Sistema | Int | | Define sistema. |
| Nota | Varchar(200) | | Define nota. |

DATOS TABLA aca_notanotas
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_notas**

Describe los datos de las notas por estudiante.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| not_codigo | varchar(20) | Primary key | Clave primaria de la tabla aca_notas. |
| mat_codigo | Int | Foreign key | Clave primaria de la tabla aca_matricula. |
| asi_codigo | int | Foreign key | Clave primaria de la tabla asignatura. |
| Seimprime | Tinyint | | Define si desea que se imprima. |
| Orden | Tinyint | | Define orden. |
| Promedia | Tinyint | | Define promedio. |
| Cuantitativa | Tinyint | | Define deforma cuantitativa. |
| Disciplina | Tinyint | | Define la disciplina. |
| sis_codigo | Int | Foreing key | Clave primaria de la tabla aca_sistemanotas |

| | | | |
|---|---|---|---|
| not_fecha | Datetime | | Define la fecha de la nota. |
| not_observacion | Varchar(200) | | Define si existe alguna observación. |
| not_disciplina | Double | | Define nota de disciplina. |
| Nota1 | Double | | Define el valor de la nota. |
| Nota2 | Double | | Define el valor de la nota. |
| Nota3 | Double | | Define el valor de la nota. |
| NotaEX | Double | | Define el valor de la nota. |
| NotaFINAL | Double | | Define el valor de la nota. |

DATOS TABLA aca_notas
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_notasfinales**

Describe las notas finales de los estudiantes de la Escuela Ciudad de Ibarra.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| fina_codigo | Varchar(20) | Primary Key | Clave primaria de la tabla aca_notasfinales |
| est_codigo | Int | Foreign key | Clave primaria de la tabla aca_estudiantes |
| fina_secuencia | Int | | Define la secuencia |
| fina_nota | Double | | Define la nota. |
| fina_estado | Varchar(1) | | Define el estado de la nota. |
| fina_tipo | Int | | Define el tipo de nota. |

DATOS TABLA aca_notasfinales
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_paralelo**

Describe el paralelo de la institución.

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|
| par_codigo | Int | Primary key | Clave primaria de la tabla aca_paralelo |
| par_nombre | varchar(30) | | Define el nombre del paralelo. |

DATOS TABLA aca_paralelo
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_parametros**

Describe los parámetros de los perfiles para el usuario.

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|
| par_secuencial | Int | Primary key | Clave primaria de la tabla aca_parametros. |
| per_codigo | Int | Foreingn key | Clave primaria de la tabla aca_periodo. |
| par_variable | Varchar(50) | | Define variable. |
| par_descripcion | Varchar(100) | | Descripción del parámetro. |
| par_tipo | Varchar(1) | | Define el tipo de parámetro. |

| | | | |
|---|---|---|---|
| par_cvalor | Varchar(100) | | Define valor. |
| par_nvalor | Double | | Define valor. |
| par_lvalor | Tinyint | | Define valor. |
| par_dvalor | Date | | Define valor. |

DATOS TABLA aca_parametros
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_pendientes**

Describe los datos pendientes.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| pen_codigo | Int | Primary key | Clave primaria de la tabla aca_pendientes |
| est_codigo | Int | Foreign key | Clave primaria de la aca_estudiantes |
| pen_mes | Int | | Define mes. |
| pen_anio | Int | | Define año. |
| pen_valor | Double | | Define valor. |

DATOS TABLA aca_pendientes
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_periodo**

Describe los datos del periodo.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| per_codigo | Int | Primary key | Clave primaria de la tabla aca_periodo. |

| | | | |
|---|---|---|---|
| ins_codigo | Int | Foreign key | Clave primaria de la tabla aca_institucion. |
| ani_codigo | Int | Foreign key | Clave primaria de la tabla aca_anio. |
| sec_codigo | Int | Foreign key | Clave primaria de la tabla aca_seccion. |
| jor_codigo | Int | Foreign key | Clave primaria de la tabla aca_jornada. |
| per_fecha_iord | Date | | Define la fecha de inicio de matrícula ordinaria. |
| per_fecha_ford | Date | | Define la fecha final de matrícula ordinaria. |
| per_fecha_iext | Date | | Define la fecha inicio de matrícula extraordinaria. |
| per_fecha_fext | Date | | Define la fecha final de matrícula extraordinaria. |
| per_estado | Tinyint | | Define el estado del periodo. |

DATOS TABLA aca_periodo
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_profesor**

Describe los datos para el usuario.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| pro_codigo | Int | Primary | Clave primaria de la |

| | | | |
|---|---|---|---|
| | | key | tabla aca_profesor. |
| pro_nombre | Varchar(50) | | Define el nombre del usuario. |
| pro_apellido | Varchar(50) | | Define el apellido del usuario. |
| pro_direccion | Varchar(60) | | Define la dirección del usuario. |
| pro_telefono1 | Varchar(13) | | Define el teléfono del usuario. |
| pro_telefono2 | Varchar(13) | | Define el teléfono del usuario. |
| pro_tipoIdentificacion | Varchar(1) | | Define la identificación del usuario. |
| pro_genero | Varchar(1) | | Define el género del usuario. |
| pro_fechaIngreso | Date | | Define la fecha de ingreso del usuario. |
| pro_estado | Tinyint | | Define el estado del usuario. |
| pro_fecha_Retiro | Date | | Define la fecha de retiro del usuario. |
| pro_email | Varchar(100) | | Define el mail del usuario. |
| pro_usuario | Varchar(13) | | Se define el usuario. |
| pro_clave | Varchar(200) | | Define la clave de usuario. |
| pro_tipoUsuario | Int | | Define el tipo de usuario. |
| pro_tratamiento | Varchar(5) | | Define el tratamiento |

| | | | |
|---|---|---|---|
| | | | que recibe el usuario. |
| per_perfil | Varchar(3) | | Define el perfil que va utilizar el usuario. |
| pro_identificacion | Varchar(13) | | Define la identificación del usuario. |
| pro_pregunta | Varchar(100) | | Define la pregunta que elige el usuario. |
| pro_respuesta | Varchar(100) | | Define la respuesta a la pregunta del usuario. |
| pro_cambioClave | Tinyint | | Define si desea el cambio de clave. |
| pro_civil | Varchar(2) | | Define el estado civil del usuario. |
| pro_sueldo | Double | | Define el sueldo del usuario. |

DATOS TABLA aca_profesor
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_seccion**

Describe las diferentes secciones que existen en la institución.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| sec_codigo | Int | Primaty key | Clave primaria de la tabla aca_seccion. |
| sec_nombre | Varchar(30) | | Define el nombre de la |

| | | | |
|---|---|---|---|
| | | | sección. |

DATOS TABLA aca_seccion
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_sistemanotas**

Describe las notas con sus respectivos periodos.

| **Columna** | **Tipo de dato** | **Detalle** | **Descripción** |
|---|---|---|---|
| sis_codigo | Int | Primary key | Clave primaria de la tabla aca_sistemanotas. |
| tip_codigo | Int | Foreign key | Clave principal de la tabla aca_tiposistema. |
| per_codigo | Int | Foreign key | Clave principal de la tabla aca_periodo. |
| sis_nombre | Varchar(100) | | Define el nombre del sistema de notas. |
| sis_abreviatura | Varchar(10) | | Define la abreviatura de la nota. |
| sis_fechaisecre | Date | | Define la fecha de ingreso de la nota. |
| sis_fechafsecre | Date | | Define la fecha final de secretaría. |
| sis_fechaiprofe | Date | | Define la fecha de inicio del profesor. |
| sis_fechafprofe | Date | | Define la fecha final del profesor. |
| sis_rangoini | Double | | Define los rangos de |

| | | | |
|---|---|---|---|
| | | | notas. |
| sis_rangofin | Double | | Define el rango. |
| sis_notalimite | Double | | Define el límite para la nota. |
| sis_conduta | Double | | Define la nota de conducta. |
| sis_seimprime | Tinyint | | Define si desea que se imprima. |
| sis_espromedio | Tinyint | | Define el promedio |
| sis_tipopromedioFin | Varchar(2) | | Define el promedio |
| sis_formula | Varchar(200) | | Define la fórmula para el cálculo de las notas. |
| sis_orden | Int | | Define el orden. |
| sis_disciplina | Tinyint | | Define si desea incluir lanota de disciplina. |
| sis_fechaidisci | Date | | Define la fecha de inicio de disciplina |
| sis_fechafdisci | Date | | Define la fecha final de disciplina |
| sis_formulaDisciplina | Varchar(200) | | Define fórmula para el cálculo de la nota de disciplina. |
| sis_verifica | Tinyint | | Define un valor que permite verificar |

DATOS TABLA aca_sistemanotas
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_tiposistema**

Describe el tipo de notas para el sistema.

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|
| tip_codigo | Int | Primary key | Clave primaria de la tabla aca_tiposistema |
| tip_nombre | Varchar(100) | | Define el nombre del tipo. |

DATOS TABLA aca_tiposistema
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

**Tabla aca_usrper**

Describe los usuarios por periodo.

| Columna | Tipo de dato | Detalle | Descripción |
|---|---|---|---|
| up_codigo | Int | Primary key | Clave primaria de la tabla aca_usrper. |
| pro_codigo | Int | Foreign key | Clave primaria de la tabla aca_profesor. |
| per_codigo | Int | Foreign key | Clave primaria de la tabla aca_periodo. |

DATOS TABLA aca_usrper
FUENTE: Tesista(s)
AUTOR: Andrea Mora, María Elisa Hernández

# ANEXO 3

MANUAL DE USUARIO

# MANUAL DE USUARIO

## INTRODUCCIÓN

El sistema **ESCIB** cuyas siglas corresponde a Escuela Fiscal Mixta Ciudad de Ibarra, es aquel que se encargara del control total de las actividades académicas que se lleva a cabo en dicha Institución.

Este manual le permitirá aprender a utilizar todas las funcionalidades básicas del sistema desarrollado para la Escuela fiscal mixta ciudad de Ibarra.

## Como acceder a ESCIB?

Existen dos maneras de ingresar al sistema:

- La primera es realizando una conexión local es decir el sistema correrá en la máquina que estamos conectados físicamente, en la cual debemos tener instalado el servidor de aplicaciones Apache 6.0, previamente subido escib.war en su carpeta principal App, además de MySQL 6.0 o superior donde se encuentre restaurada la base de datos.

  Bajo estas indicaciones en nuestro navegador digitamos http//localhost:8084/escib el cual nos llevará al inicio de nuestra aplicación, como se muestra en la Fig. 1.

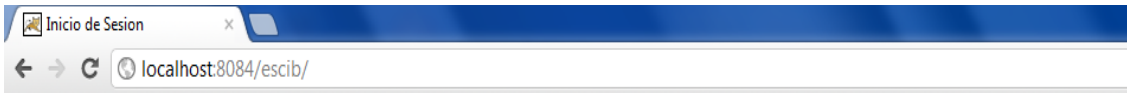


Fig. 1 Ingreso a la aplicación

Cabe recalcar que el puerto 8084 puede variar dependiendo de la configuración de nuestro navegar web.

- La segunda opción para conectarnos a nuestro sistema es bajo un esquema cliente servidor; en el cual la máquina desde la que accedemos será cliente y la máquina donde se encuentre alojada nuestra aplicación e instalado el servidor de aplicaciones junto con la base de datos será el servidor.

Para poder acceder al sistema a través de nuestro navegador tenemos que conocer la dirección física IPV4 del servidor donde se encuentra alojado **escib.war;** en nuestro caso es http//192.168.0.101:8084/escib con esta línea digitada en nuestro navegador nos llevara al inicio de sesión de la aplicación, como se muestra en la Fig.2.


Fig. 2 Ingreso a la aplicación

## 1. INICIO DE SESIÓN

Una vez en el sistema la pantalla principal de inicio de sesión será la siguiente:

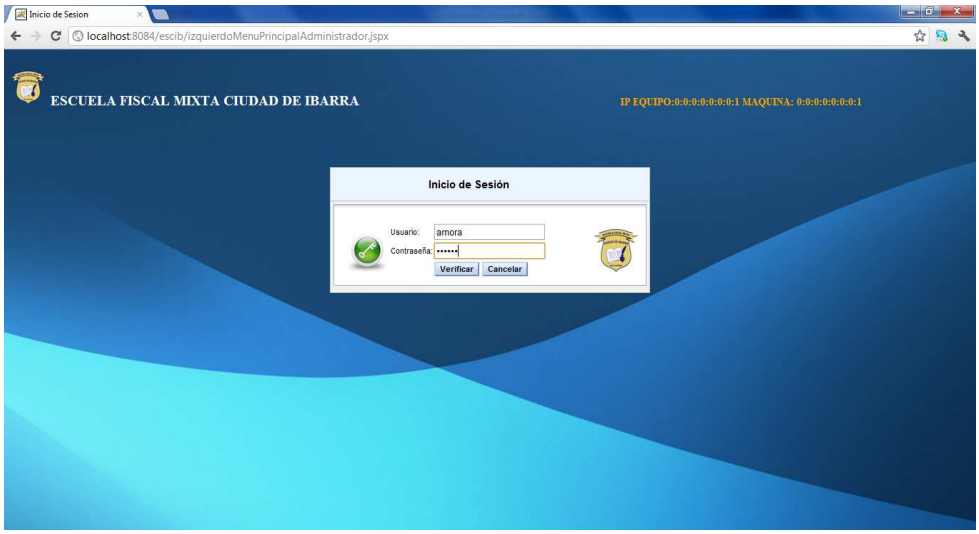



Fig. 3 Inicio de sesión

Como indica la Fig.3 hay que ingresar el usuario y contraseña proporcionados por el administrador del sistema; cabe recalcar que cuando se realiza una conexión local la dirección IP y la máquina por defecto toma la dirección 0.0.0.1 respectivamente. Cuando nos conectemos por la modalidad cliente servidor el sistema recuperará la IP y usuario de la máquina con la que accedemos a **ESCIB**.

Para continuar damos clic en verificar, como indica la Fig. 4.

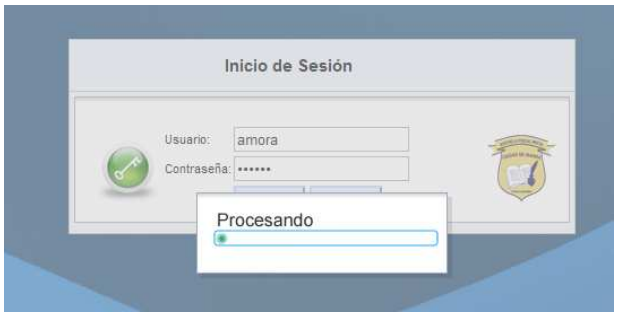



Fig. 4 Logueo aplicación

## 2. SELECCIÓN PERIODO ACADÉMICO

En la Fig.5 se observa una lista con el nombre del colegio y los periodos que se encuentran configurados para dicho usuario, los periodos corresponden a los años lectivos disponibles; en esta pantalla parecida al inicio de sesión se selecciona el periodo académico a acceder.

Aquí se encuentran todos los periodos configurados para el usuario que ha iniciado la sesión local o cliente servidor.

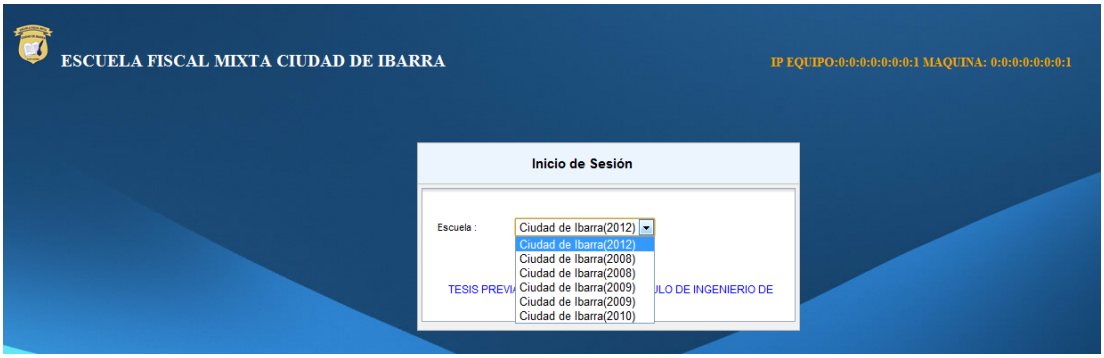



Fig.5 Periodo académico

Seleccionar un año lectivo y dar clic en iniciar.

## 3. MENU PRINCIPAL

En el menú principal se encuentran los módulos que maneja el sistema; con una barra informativa de la institución, el sello, el periodo previamente seleccionado y el usuario que se conectó al sistema para mantener un control del mismo.

## 4. MÓDULOS DEL SISTEMA

La Fig.6 indica la pantalla con los Módulos del sistema.

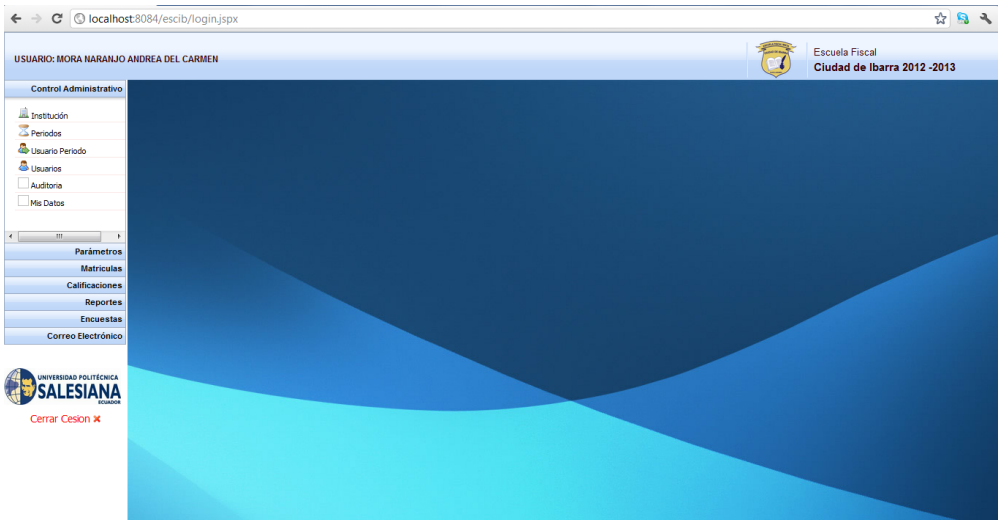



Fig. 6 Módulos del sistema

### 4.1 Control administrativo

Control administrativo es un módulo en el cual se puede cambiar la configuración de la escuela, los años lectivos además permite anclar cada usuario a un periodo único, a varios periodos o a todos. Como administrador se puede acceder a auditorías que es una tabla de todas las transacciones que se han llevado a cabo para dicha configuración. El módulo de control administrativo posee en su lista de propiedades la sección Mis datos en el cual se puede modificar la información del usuario, como indica la Fig.7.

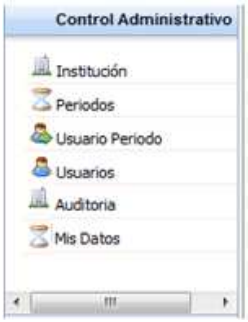



Fig. 7 Control administrativo

### 4.1.1. INSTITUCIÓN

En esta sección del módulo control administrativo corresponde a la información general de la escuela; en la cual previamente se debe tener cargado los datos para poder acceder; en el formulario inicial nos da la opción de ingresar siempre y cuando no exista configurada nuestra institución, en nuestro caso podemos realiza cambios seleccionando la escuela de la tabla inferior y dando clic sobre editar.

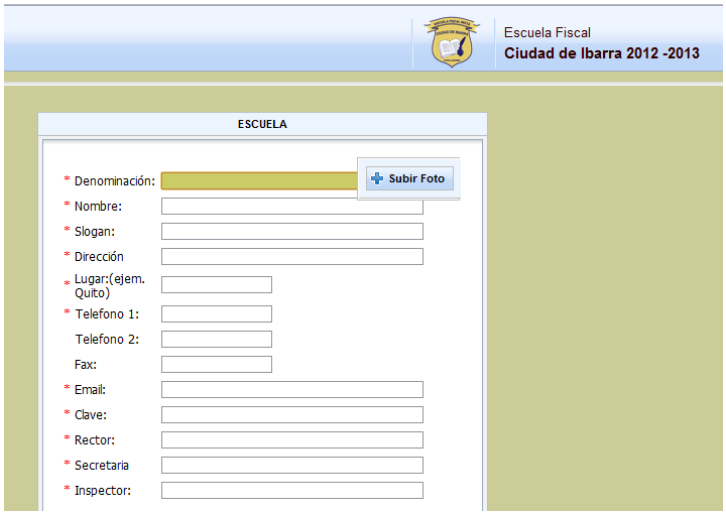



Fig. 8 Institución

Los campos que se encuentran marcados con * de color rojo son mandatorios no pueden quedar sin información.

### 4.1.2 PERIODOS

En esta sección del módulo se configurar los años lectivos, el inicio de inscripciones, finalización de la misma e inicio y finalización del proceso de matriculación para cada año lectivo.

De igual manera que la sección anterior se ingresar un nuevo periodo llenando los datos correspondientes o editar información seleccionando

los periodos ya configurados antes; dando clic en el botón editar de la tabla junto a cada registro, como se muestra en la Fig.9.

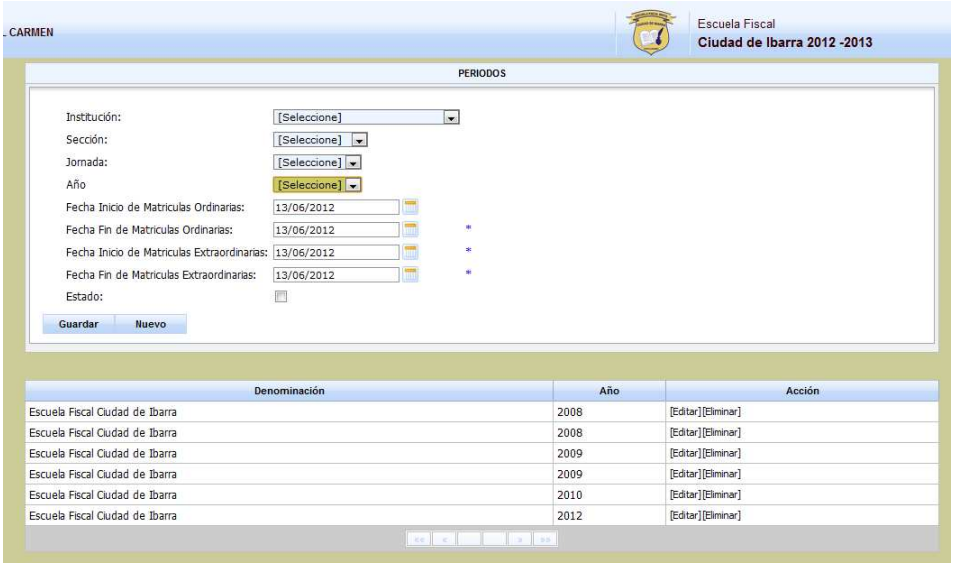



Fig. 9 Periodos

### 4.1.3 USUARIO PERIODO

Se asigna a cualquier profesor para que sea parte de algún año lectivo; además en la tabla inferior del formulario se tiene todos los profesores o administradores configurados inicialmente para dichos periodos; los cuales igual que las secciones anteriores se da un clic en quitar para eliminar de cada año lectivo, como se indica en la Fig.10.

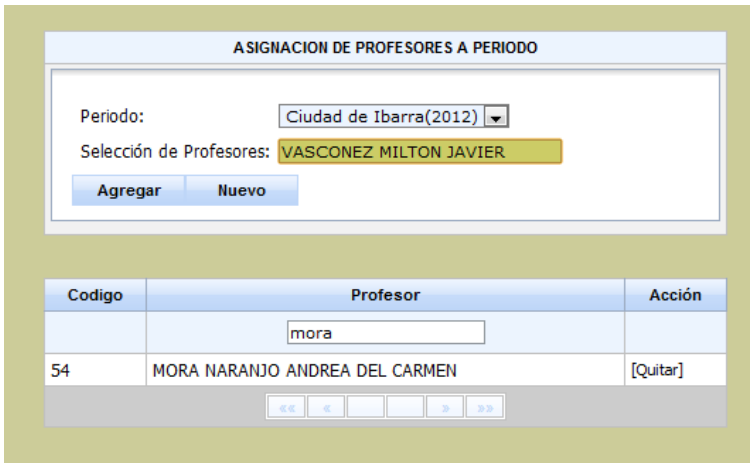



Fig. 10 Usuario por periodo

### 4.1.4 USUARIOS

Esta es la sección principal de usuarios en la cual se crear nuevos usuarios desde 0 al cual le podemos dar las propiedades de administrador, inspector, profesor o secretaria; a través de los cuales podrá navegar por el sistema según los permisos o privilegios configurados.

Inicialmente asignar uno o varios años lectivos, los cuales se muestra en la segunda pantalla de selección previo al ingreso del sistema, como indica la Fig. 11.

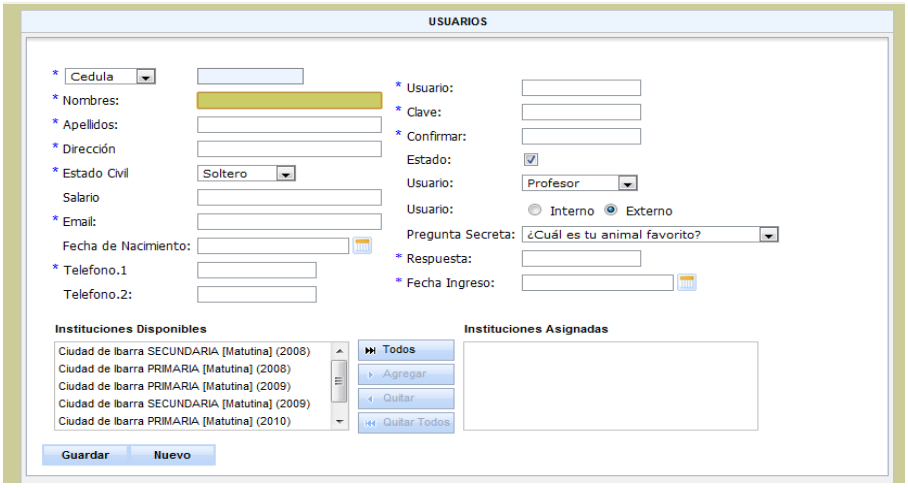
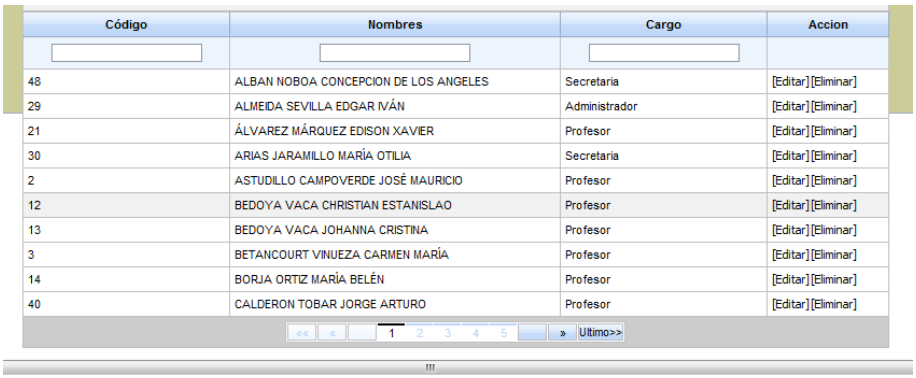



Fig.11Usuarios

Se debe tener presente los campos marcados con asterisco (*) son obligatorios y llenar en caso de ingreso de un nuevo registro o edición del mismo.

Al igual que en los diferentes módulos del sistema la sección usuarios permite seleccionar un registro de la tabla inferior para realizar cualquier tipo de modificación al usuario seleccionado, habilitarlo, pregunta secreta con la cual podrá recuperar su contraseña, como indica la Fig. 12.



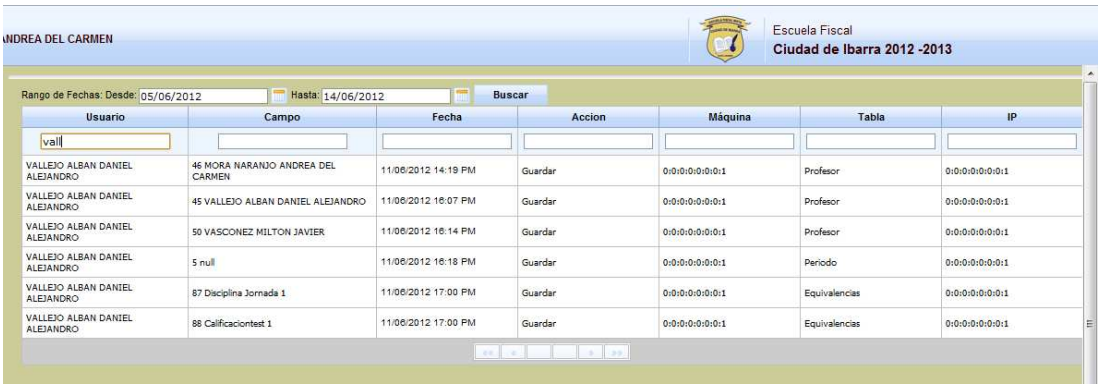
| Código | Nombres | Cargo | Accion |
|---|---|---|---|
| 48 | ALBAN NOBOA CONCEPCION DE LOS ANGELES | Secretaria | [Editar] [Eliminar] |
| 29 | ALMEIDA SEVILLA EDGAR IVÁN | Administrador | [Editar] [Eliminar] |
| 21 | ÁLVAREZ MÁRQUEZ EDISON XAVIER | Profesor | [Editar] [Eliminar] |
| 30 | ARIAS JARAMILLO MARÍA OTILIA | Secretaria | [Editar] [Eliminar] |
| 2 | ASTUDILLO CAMPOVERDE JOSÉ MAURICIO | Profesor | [Editar] [Eliminar] |
| 12 | BEDOYA VACA CHRISTIAN ESTANISLAO | Profesor | [Editar] [Eliminar] |
| 13 | BEDOYA VACA JOHANNA CRISTINA | Profesor | [Editar] [Eliminar] |
| 3 | BETANCOURT VINUEZA CARMEN MARÍA | Profesor | [Editar] [Eliminar] |
| 14 | BORJA ORTIZ MARÍA BELÉN | Profesor | [Editar] [Eliminar] |
| 40 | CALDERON TOBAR JORGE ARTURO | Profesor | [Editar] [Eliminar] |

1 2 3 4 5 » Ultimo>>

Fig. 12 Elección usuarios

### 4.1.5 AUDITORIA

En esta sección se muestra las transacciones realizadas por los usuarios, con un control por fecha, acción, campo, tabla, maquina e Ip; a través de este control se verificar las transacciones de uno u otro usuario dentro del sistema; cabe recalcar que no es posible eliminar la información que se registre en el mismo; a menos que tengan algún tipo de acceso y privilegios a la base de datos, como muestra la Fig. 13.



ANDREA DEL CARMEN

Escuela Fiscal
Ciudad de Ibarra 2012 -2013

Rango de Fechas: Desde: 05/06/2012 Hasta: 14/06/2012 Buscar

| Usuario | Campo | Fecha | Accion | Máquina | Tabla | IP |
|---|---|---|---|---|---|---|
| vall | | | | | | |
| VALLEJO ALBAN DANIEL ALEJANDRO | 46 MORA NARANJO ANDREA DEL CARMEN | 11/06/2012 14:19 PM | Guardar | 0:0:0:0:0:0:0:1 | Profesor | 0:0:0:0:0:0:0:1 |
| VALLEJO ALBAN DANIEL ALEJANDRO | 45 VALLEJO ALBAN DANIEL ALEJANDRO | 11/06/2012 16:07 PM | Guardar | 0:0:0:0:0:0:0:1 | Profesor | 0:0:0:0:0:0:0:1 |
| VALLEJO ALBAN DANIEL ALEJANDRO | 50 VASCONEZ MILTON JAVIER | 11/06/2012 16:14 PM | Guardar | 0:0:0:0:0:0:0:1 | Profesor | 0:0:0:0:0:0:0:1 |
| VALLEJO ALBAN DANIEL ALEJANDRO | 5 null | 11/06/2012 16:18 PM | Guardar | 0:0:0:0:0:0:0:1 | Periodo | 0:0:0:0:0:0:0:1 |
| VALLEJO ALBAN DANIEL ALEJANDRO | 87 Disciplina Jornada 1 | 11/06/2012 17:00 PM | Guardar | 0:0:0:0:0:0:0:1 | Equivalencias | 0:0:0:0:0:0:0:1 |
| VALLEJO ALBAN DANIEL ALEJANDRO | 88 Calificaciontest 1 | 11/06/2012 17:00 PM | Guardar | 0:0:0:0:0:0:0:1 | Equivalencias | 0:0:0:0:0:0:0:1 |

Fig. 13 Auditoría

### 4.1.6 MIS DATOS

Aquí se tiene la información del docente ya que como profesor no tiene acceso a todos los usuarios; en esta pantalla podrá modificar su información personal los campos que el sistema le permita, como indica la Fig.14.

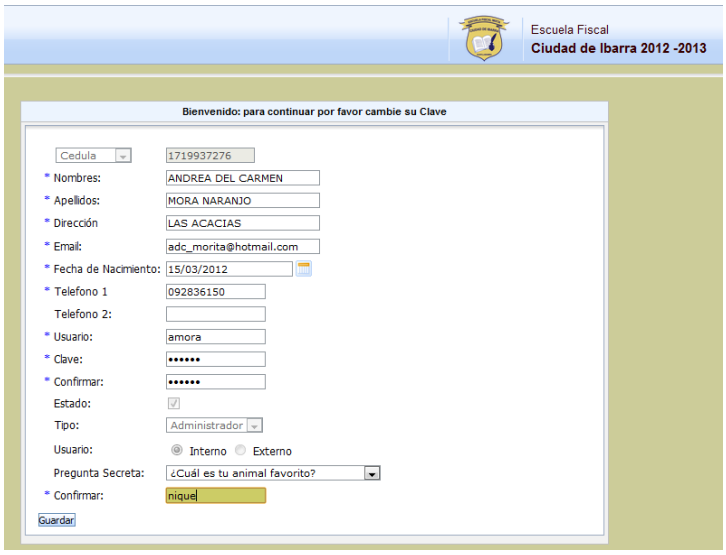



Fig. 14 Mis datos

## 4.2 PARÁMETROS

### 4.2.1 ASIGNATURAS

En este módulo del sistema realiza la creación, edición o eliminación de asignaturas las cuales pueden ser visibles para cualquier años lectivo; es decir se configura una sola vez; en este módulo tenemos nombre de la asignatura, abreviatura, y si es de tipo aprovechamiento o disciplina.

Al igual que en los diferentes módulos se selecciona de la tabla inferior las asignaturas ya configuradas para editarlas o eliminarlas.

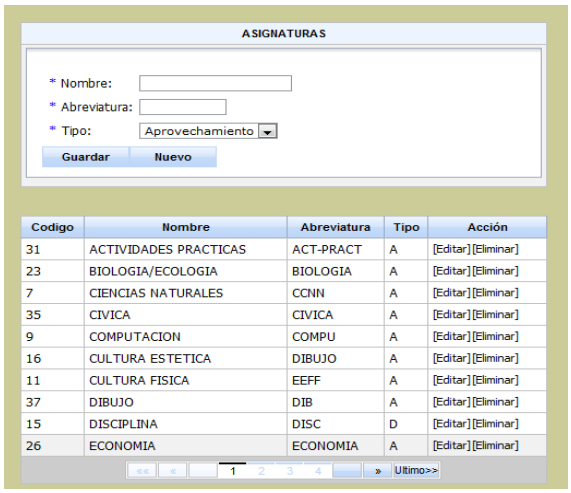



Fig. 15 Asignaturas

### 4.2.2 AÑOS

En este módulo se lleva un control total sobre los años para los periodos lectivos correspondientes, en el cual se insertar, edita o eliminar años, siempre y cuando no existan transacciones generadas en dicho año, como la Fig. 16.

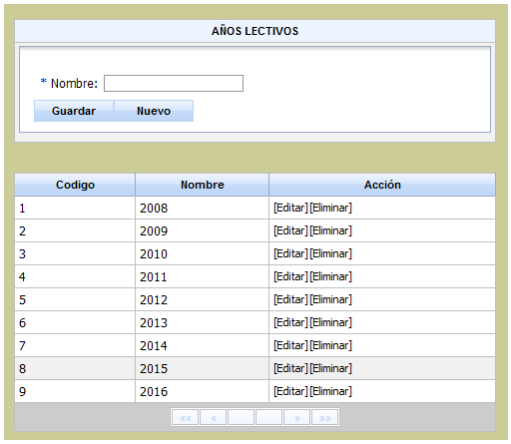



Fig. 16 Años

### 4.2.3 Cursos

En la sección cursos del módulo de parámetros, es donde asignamos los paralelos para cada nivel, así como los cupos por cada curso, la nota mínima, la nota máxima, el inspector el dirigente; además permitirá seleccionar de la tabla de cursos para poder realizar actualizaciones o eliminación de registros; siempre y cuando no tengan líneas de transacciones para dicho curso, como indica la Fig.17.

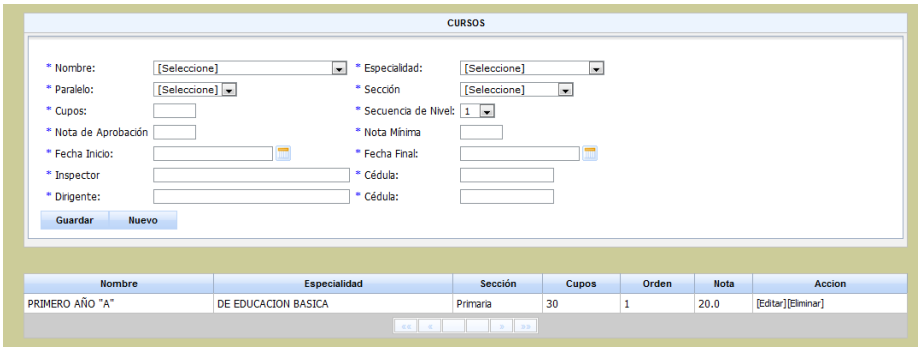



Fig. 17 Cursos

### 4.2.4 Nombres Cursos

El módulo de nombres curso es aquel que permite ingresar el nombre con el que se definir, para ser anclado posteriormente a un paralelo y nivel, como muestra la Fig. 18.
Permite insertar un nuevo registro, editarlo o eliminarlo, tomar en cuenta que nos existan transacciones para dicho nombre del curso.


Fig. 18 Nombre cursos

### 4.2.5 PARALELOS

Sección del módulo de parámetros el cual permite ingresar la sigla de los paralelos que van a existir dentro del sistema y que pueden ser anclados a los diferentes cursos y niveles, como indica la Fig.19.

Al finalizar la edición o creación damos clic en guardar.

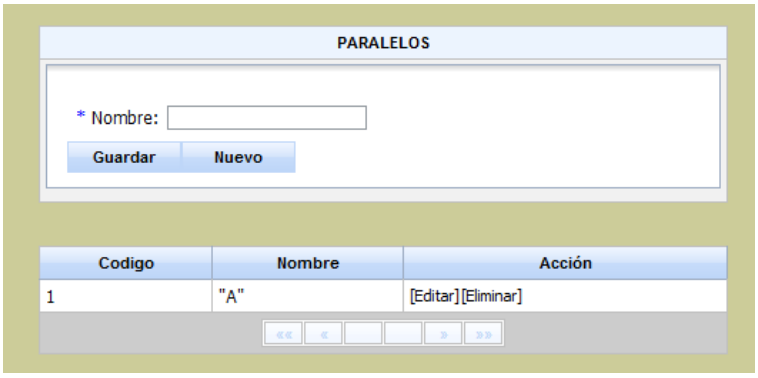



Fig.19 Paralelos

### 4.2.6 PERMISOS

Aquí se maneja la permisiología del sistema, en este módulo no agregamos registros pues ya se encuentran definidos los módulos; aquí únicamente se asigna a cada tipo de usuario permisos de

lectura, escritura, edición o eliminación para cada módulo dependiendo del tipo de usuario, como muestra la Fig. 20; al final de la parametrización de permisos damos clic en guardar.

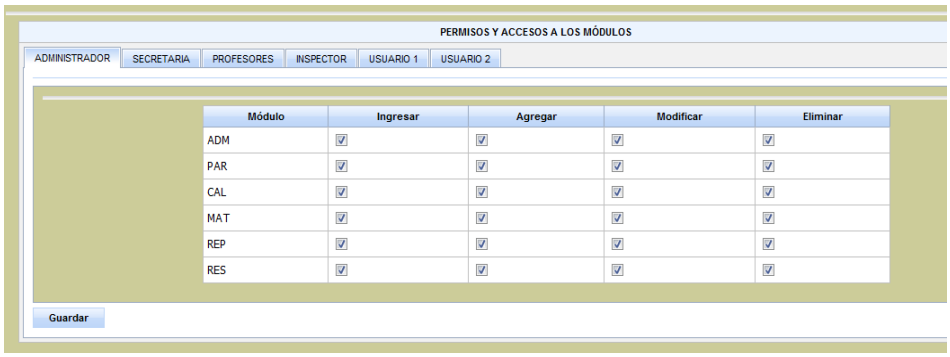



Fig.20 Permisos

# ANEXO 4

MANUAL TÉCNICO

# MANUAL TÉCNICO

1.- Se debe verificar que se encuentre levantado el servidor de aplicaciones Apache Tomcat 6.0, como se muestra en la Fig.1.

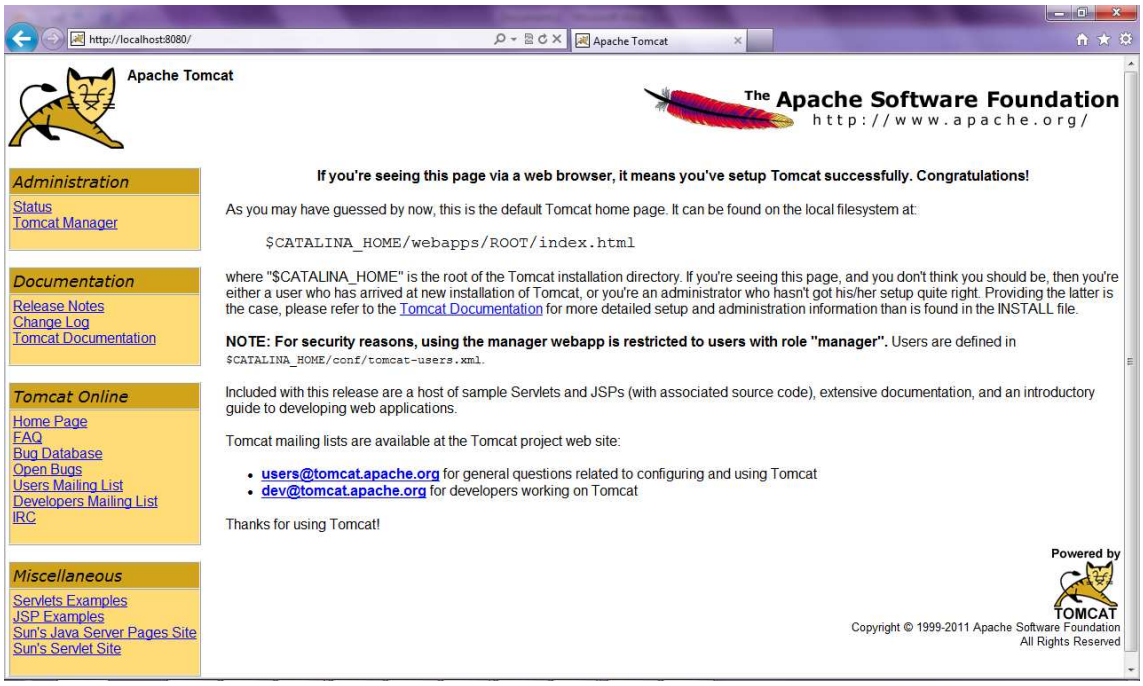



Fig. 1 Verificar Apache Tomcat

2.- Publicar la aplicación dentro de la carpeta webapp del servidor de aplicaciones, como se muestra en la Fig.2.

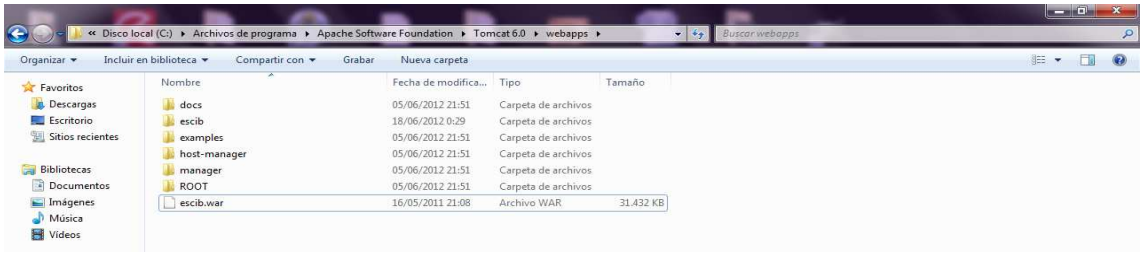



Fig.2 Publicación de aplicación

3.- Verificar la instalación de MySql 6.0, MySql Administrator que vaya de acuerdo a la versión del motor de base de datos.

4.- Verificar previamente a la restauración de la base de datos que el usuario y la contraseña sean root respectivamente.

5.- Iniciar el servidor de apache tomcat 6.0

6.- Iniciar el servicio de Apache Tomcat como se muestra en la Fig. 3 desde Panel de control\Todos los elementos de Panel de control\Herramientas Administrativas

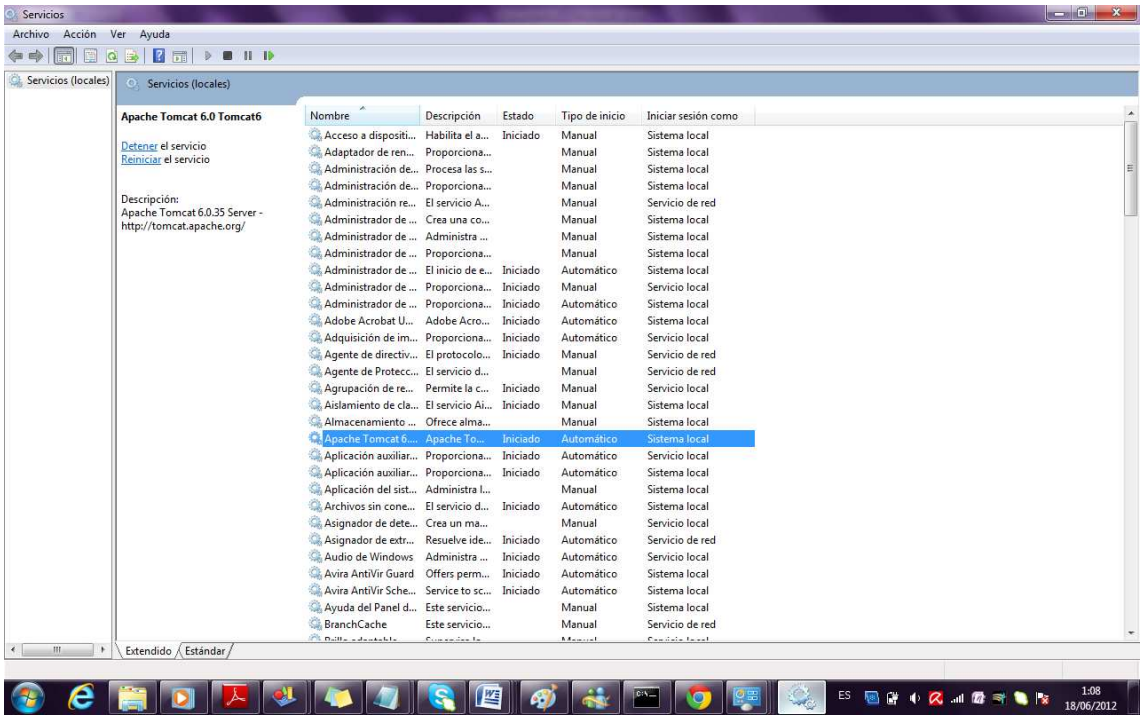



Fig. 3 Iniciar servicio

7.- Para validar ingresar a un navegador de preferencia Google Chrome y digitar la siguiente dirección htpp://192.168.4.1/escib, como se muestra en la Fig. 4.

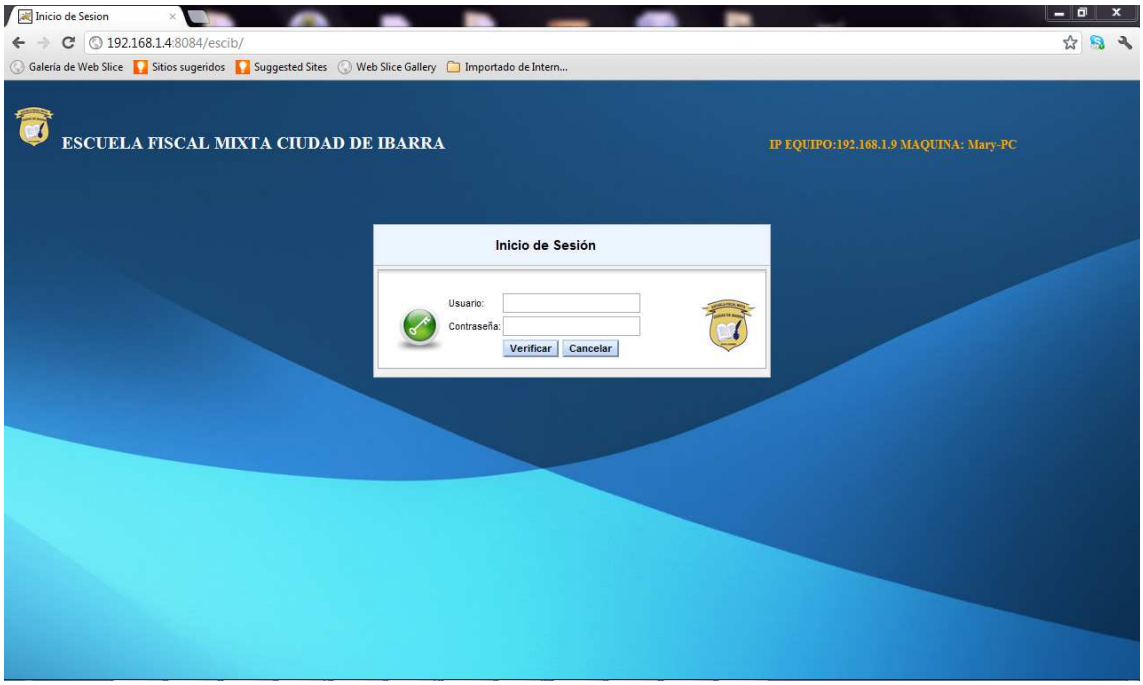



Fig. 4 Ingresar a la aplicación